Term Rewriting Systems and the Church-Rosser Property

by

Yoshihito Toyama

May 1990

Abstract

Our research concerns term rewriting systems having the Church-Rosser property. First, we develop modular aspects for term rewriting systems for systematic analysis of the behavior of combined term rewriting systems. The main outcome of this research is establishment of the modularity of the Church-Rosser property, which is very helpful for dealing with complex term rewriting systems.

We also develop a simple method for proving the equivalence of two given term rewriting systems without the explicit use of induction, and demonstrate that the method can be effectively applied to deriving a new term rewriting system from a given one by using equivalence transformation rules.

Finally, the Church-Rosser property of a new type of conditional term rewriting systems is investigated.

Acknowledgments

I would like to thank Professor Akira Maruoka of Tohoku University for his kindly guidance and encouragement during the preparation of this thesis. I would also like to thank Professor Masayuki Kimura, Professor Takayasu Ito, and Professor Masahiko Sato of Tohoku University for their helpful suggestions and comments.

I wish to express my thanks to NTT (Nippon Telegraph and Telephone Corporation) Laboratories for giving a rich and stimulating environment for my research. Especially thanks to Dr. Kazuhiko Kakehi, executive manager of Information Science Research Laboratory, and the members of Theory Group - Dr. Kenji Koyama, Mr. Hirofumi Katsuno, Mr. Yasuyoshi Okada, Mr. Junnosuke Yamada, and Mr. Ken Mano - for their kindly supports and encouragements.

I also wish to thank Professor Jan Willem Klop, Center for Mathematics and Computer Science, and Professor Henk Pieter Barendregt, Nijmegen University, for valuable discussions. The content of Section 2.3, *Termination for the Direct Sum of Left-Linear Complete Term Rewriting Systems*, is based on a collaborative work with them.

Contents

1. Introduction

Our research concerns term rewriting systems having the Church-Rosser property [22, 27, 39, 58]. We investigate the modular structure of term rewriting systems, which is clearly important in building a complex system from simple parts. Modularity for the direct sum and union of term rewriting systems are developed for systematic analysis of the behavior of combined systems. This research establishes the modularity of the Church-Rosser property, which is very useful in dealing with complex term rewriting systems.

The equivalence problem frequently appears in computer science [9]. We develop a simple method for proving the equivalence of two given term rewriting systems without the explicit use of induction. The method extends the *inductionless induction* methods developed by Musser [69], Goguen [31], Huet and Hullot [38] into a more general framework. We demonstrate that the method can be effectively applied to deriving a new term rewriting system from a given one by using equivalence transformation rules.

Finally, the Church-Rosser property of a new type of conditional term rewriting systems is investigated.

1.1. Backgrounds

Equational reasoning has been applied to many problems in computer science. A number of effective methods for handling equations in automated theorem proving have been proposed [11, 35, 52, 59, 62, 71]. Many formula manipulation systems use

equations to simplify a given expression [8]. In the field of algebraic specifications, data types are axiomatized in equational theories [30, 31, 33, 84]. Several functional and logic programming languages also base their semantics on equational logics; thus, equational reasoning plays a central role in computation, program verification, program transformation, and program synthesis [9, 19, 40, 41, 42, 43, 72, 73].

In reasoning about equations, we need to decide whether an equation is a consequence of a given set of equations (axioms). As well known, this reasoning is usually done by replacing equals with equals. However, dealing effectively with this equational reasoning is, in general, difficult. Instead, an asymmetrical reasoning, *reduction*, has been proposed, which uses equations in only one direction to rewrite a term into a simpler form. For instance, an equation $1+2=3$ can be interpreted as "3 is the result of of computing $1+2$", but not vice versa. This directional replacement is expressed by $1+2 \rightarrow 3$ which reads "$1+2$ reduces to 3". This computational aspect for equations naturally leads us to term rewriting systems [22, 27, 39, 58].

Term rewriting systems are sets of directed equations (rewriting rules) used to compute by repeatedly replacing equal terms in a given formula until the simplest form possible (normal form) is obtained. The Church-Rosser property is certainly one of the most fundamental properties of term rewriting systems (and the reduction paradigm). In a Church-Rosser term rewriting system, the simplest form of a given term is unique, that is, the final result does not depend on the order in which the rewriting rules were applied. This flexibility permits us to compute the result in an asynchronous parallel or nondeterministic way. Furthermore, if every reduction always terminates, a Church-Rosser term rewriting system determines a decision procedure for the word problem for the corresponding equational theory. Thus, Church-Rosser systems can offer both flexible computing and effective reasoning with equations, and have been intensively researched and widely applied to automated theorem proving, functional and logic programming, algebraic specification, program verification, program transformation, program synthesis, and symbolic computation [9, 8, 11, 19, 30, 31, 33, 35, 52, 59, 62, 71, 72, 73, 84].

Sufficient criteria for proving the Church-Rosser property have been discovered [59, 81, 72, 37] and many applications have been developed within these criteria. However, the criteria requires tight limitations on systems, such as termination [59] or left-linearity [81, 72, 37]. Few criteria have been proposed that do not have these limitations. Thus, it is worth while to extend criteria for the Church-Rosser property and exploit new potentials of term rewriting systems having this property.

We will here give a short sketch of the history of term rewriting systems. The study of term rewriting systems originated in combinatory logic [15, 16] and lambda calculus [2, 12], which were developed and deeply analyzed half a century ago to investigate the foundation of functions. Combinatory logic is actually a term rewriting system.

Combinatory logic was invented independently in the 1920's by Schöenfinkel [86] and Curry [15, 16]. The aim was to do logic or computation without using variables. With a different motivation, lambda calculus was developed in the 1930's by Church [12], which is a variant of combinatory logic. Kleene showed that lambda-definability is equivalent to "effective computability" defined by partial recursive functions [55]. Rosser demonstrated the close connection between lambda calculus and combinatory logic [82], and Church and Rosser showed the consistency of both systems by using the Church-Rosser property [13]. Through these early studies, most of the basic concepts and important techniques in the field of term rewriting systems were introduced.

Recursive program schemata, a special case of term rewriting systems, were introduced in the 1960's by McCarthy to study the computation of recursively defined functions [63]. In the 1970's, considering recursive program schemata as rewriting systems, Vuillemin [106, 107], Downey and Sethi [24], Berry and Lévy [4] developed a mathematical framework for the operational semantics of LISP-like programming languages, and they investigated reduction strategy problems as the correct or optimal implementation problems of the languages. Rosen [81], O'Donnel [72], and

Huet [37] extended this framework into a theory of *non-terminating* the Church-Rosser term rewriting systems and proposed a few sufficient criteria for the Church-Rosser property of left-linear systems. Moreover, O'Donnel[72, 73] suggested an equational programming paradigm based on term rewriting systems, and in the 1980's, functional programming languages in which programs can be written as a set of equations, such as Miranda [105], KRC [104], HOPE [10] and T [40], have been implemented. Recently, Dershowitz and Plaisted [21] showed that conditional term rewriting systems [48] provide a paradigm for programming languages combining functional programming with logic programming, such as EQLOG [32], SLOG [25], and Qute [85].

An important application of the Church-Rosser term rewriting systems to the field of automated theorem proving, the completion procedure, was begun in 1970 by Knuth and Bendix [59]. The completion procedure takes a set of equations as input, and attempts to produce as output a complete (i.e., Church-Rosser and terminating) term rewriting system, which determines a decision procedure for the word problem for the theory. Considering *critical pairs* between rewriting rules, Knuth and Bendix demonstrated a necessary and sufficient criterion for the Church-Rosser property of terminating term rewriting systems. Variants of the basic Knuth-Bendix completion procedure (REVE [62], RRL [52], Metis [71], HIPER [11], etc.) have been developed for automated theorem proving.

1.2. Organization

This thesis includes four studies of term rewriting systems having the Church-Rosser property, arranged into Chapters 2-5. Each chapter has its own introduction and conclusion. The chapters can thus be read independently after this chapter. We give a short summary of the thesis.

The Church-Rosser property is one of the most important properties of term

rewriting systems, and sufficient criteria for proving this property have been studied [59, 81, 72, 37]. A necessary and sufficient criterion for the Church-Rosser property of terminating term rewriting systems, in which every reduction must terminate, was demonstrated by Knuth and Bendix [59]. For non-terminating term rewriting systems, Rosen [81] proved that left-linear and non-overlapping term rewriting systems (i.e., no variable occurs twice or more in the left-hand side of a rewriting rule and two left-hand sides of rewriting rules must not overlap) are Church-Rosser, and the non-overlapping limitation was somewhat relaxed by Huet [37]. However, few criteria have been proposed for the Church-Rosser property of term rewriting systems not having these limitations. Thus, it is worth while extending the above criteria. Chapters 2 and 3 demonstrate very useful facts for extending the criteria for the Church-Rosser property and for the related basic property. The key idea behind these chapters is modular properties of term rewriting systems which enable us to build a complex system from its simple parts.

In Chapter 2, we introduce the concept of the direct sum of term rewriting systems and demonstrate modular properties. The main result, which is proven in the first half of this chapter, is that the Church-Rosser property is modular: term rewriting systems R_1 and R_2 are Church-Rosser iff the direct sum $R_1 \oplus R_2$ is so. Here, the direct sum means the union of systems having disjoint sets of function symbols.

The first study on the direct sum system was conducted by Klop in [56] in order to consider the Church-Rosser property for combinatory reduction systems having non-linear rewriting rules. He showed that if R_1 is a left-linear and non-overlapping system and R_2 consists of the single non-linear rule $D(x, x) \triangleright x$, then the direct sum $R_1 \oplus R_2$ has the Church-Rosser property. However, the restriction on R_1 plays an essential role in his proof of the Church-Rosser property of $R_1 \oplus R_2$; hence his result cannot be applied to term rewriting systems without this restriction. Moreover, Klop's direct sum, which is different from ours, does not preserve the Church-Rosser

property [56].

In contrast with Klop's, our direct sum of term rewriting systems R_1 and R_2, independent of their properties such as linear or non-overlapping, always preserves their Church-Rosser property. Thus, the modularity presented here offers very useful method for proving the Church-Rosser property of combined systems without these limitations.

The second half of this chapter proves that the completeness (i.e., Church-Rosser and termination) property of term rewriting systems is modular under left-linearity: term rewriting systems R_1 and R_2 are left-linear and complete iff the direct sum $R_1 \oplus R_2$ is so. To appreciate the non-triviality of this modularity, it is contrasted with the facts that other fundamental properties, termination and completeness, are not modular [96]. The modularity presented is of value not only because it establishes a result that in itself is simple enough, but also because of the analysis necessary for the proof which gives a kind of structure theory for disjoint combinations of term rewriting systems.

Hindley [34] and Rosen [81] showed that if R_1 and R_2 commute and have the Church-Rosser property, then the union $R_1 \cup R_2$ also has the Church-Rosser property. Thus, without the requirement of the direct sum, commutativity also offers modularity for the union $R_1 \cup R_2$ of two systems. Simple sufficient criteria for commutativity or quasi-commutativity of left-linear term rewriting systems R_1 and R_2 have been proposed [1, 44, 46, 79, 88]. However, these works were done on the restrictions: R_1 and R_2 are non-overlapping with each other [1, 79, 88], or R_1 is $(E-)$ terminating [44, 46]. In Chapter 3, we study commutativity of left-linear term rewriting systems R_1 and R_2 without the above restrictions. That is, two systems may overlap and be non-terminating. It is shown that our result can be applied to proving the Church-Rosser property of left-linear term rewriting systems to which the sufficient criteria proposed by Knuth and Bendix [59], Rosen [81], and Huet [37] cannot directly apply.

Chapters 4 and 5 are not related to modularity of term rewriting systems, but investigate two topics about term rewriting systems having the Church-Rosser property. The concept of equivalence in a restricted domain of term rewriting systems frequently appears in computer science: automated theorem proving, semantics of functional programs, program transformation, verification of programs, and specification of abstract data types. However, this equivalence cannot generally be proved by mere equational reasoning; some kind of induction on the domain structure is necessary. Chapter 4 presents a new simple method for proving the equivalence in a restricted domain of two term rewriting systems without the explicit use of induction. Our approach to this problem was inspired by the *inductionless induction* methods developed by Musser [69], Goguen [31], Huet and Hullot [38], and others [26, 45, 50, 53, 60, 75, 76, 89]. We generalize the *inductionless induction* methods within a general framework. The point of our method is that the equivalence in a restricted domain can be easily proved by the Church-Rosser property and reachability. Some limitations of the *inductionless induction* methods are removed. Furthermore, we demonstrate that the extended *inductionless induction* concept is a useful tool for proving correctness of program transformations proposed by Burstall and Darlington [9].

In Chapter 5, we propose a new type of conditional term rewriting systems: the membership-conditional term rewriting system, in which each rewriting rule may have membership conditions. Our idea of membership-conditional rewriting was inspired by Church's non-linear δ-rule in λ-calculus [2, 56]. It is well known that the membership conditions in Church's δ-rule play an important role for the Church-Rosser of λ-calculus with the non-linear δ-rule. We extend the idea of membership-conditional rewriting offered in Church's δ-rule to non-linear term rewriting systems. We discuss the sufficient criteria for the Church-Rosser property of membership-conditional term rewriting systems that are non-terminating and non-linear.

Chapter 6 summarizes our research.

1.3. Reduction Systems

In this and next sections we present elementary notions and facts necessary for understanding the following chapters. Basic concepts and properties of term rewriting systems can be stated more generally in an abstract framework. Thus we start with (abstract) reduction systems, which are no more than sets equipped with some binary relations. In the next section, the abstract reductions are specialized to *rewritings* of terms. The reader who is familiar with term rewriting systems can skip these chapters. Most of the material in Sections 1.3 and 1.4 is standard and can be found in the surveys: Dershowitz and Jouannaud [22], Futatsugi and Toyama [27], Huet and Oppen [39], Klop [58].

We define reduction systems and state some simple facts about them, according to Huet [37], Klop [56, 58], and Rosen [81]. Since these reduction systems have only an abstract structure, they are called abstract reduction systems in Klop [58]. (They are also called replacement systems in Staple [87], and general replacement systems in Rosen [81].)

1.3.1. Definition

(i) A reduction system is a structure $R = \langle A, \rightarrow \rangle$ consisting of some object set A and some binary relation $\rightarrow$ on A (i.e., $\rightarrow \subseteq A \times A$), called a reduction relation. A reduction (starting with x_0) in R is a finite sequence $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \cdots \rightarrow x_n$ or an infinite sequence $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \cdots$.

(ii) The identity of elements x, y of A (or syntactical equality) is denoted by $x \equiv y$. $\overset{=}{\rightarrow}$ is the reflexive closure of $\rightarrow$, $\overset{+}{\rightarrow}$ is the transitive closure of $\rightarrow$, $\overset{*}{\rightarrow}$ is the transitive reflexive closure of $\rightarrow$, and $=$ is the equivalence relation generated by $\rightarrow$ (i.e., the transitive reflexive symmetric closure of $\rightarrow$). $\overset{m}{\rightarrow}$ denotes a

reduction of m ($m \geq 0$) steps.

(iii) If $x \in A$ is minimal with respect to $\rightarrow$, i.e., $\neg\exists y \in A[x \rightarrow y]$, then we say that x is a normal form, or $\rightarrow$ normal form; let NF be the set of normal forms. If $x \xrightarrow{*} y$ and $y \in NF$ then we say x has a normal form y and y is a normal form of x.

1.3.2. Definition.

(i) $R = \langle A, \rightarrow \rangle$ is strongly normalizing (denoted by $SN(R)$), or R is terminating, iff every reduction in R terminates, i.e., there is no infinite sequence $x_0 \rightarrow x_1 \rightarrow x_2 \rightarrow \cdots$.

(ii) R is weakly normalizing (denoted by $WN(R)$) iff any $x \in A$ has a normal form.

1.3.3. Definition. $R = \langle A, \rightarrow \rangle$ has the Church-Rosser property (denoted by $CR(R)$), or R is confluent, iff $\forall x, y, z \in A[x \xrightarrow{*} y \wedge x \xrightarrow{*} z \Rightarrow \exists w \in A, y \xrightarrow{*} w \wedge z \xrightarrow{*} w]$.

We express this property with the diagram in Figure 1.1. In this sort of diagram, dashed arrows denote (existential) reductions depending on the (universal) reductions shown by full arrows.

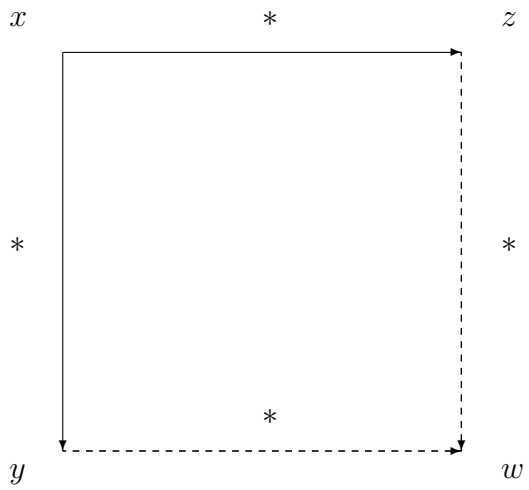


Figure 1.1.

The following proposition is well known [2, 37, 58].

1.3.4. Proposition. Let R have the Church-Rosser property, then,

(i) $\forall x, y \in A[x = y \Rightarrow \exists w \in A, x \xrightarrow{*} w \wedge y \xrightarrow{*} w]$,

(ii) $\forall x, y \in NF[x = y \Rightarrow x \equiv y]$,

(iii) $\forall x \in A \forall y \in NF[x = y \Rightarrow x \xrightarrow{*} y]$.

Let $R_1 = \langle A, \underset{1}{\rightarrow} \rangle$ and $R_2 = \langle A, \underset{2}{\rightarrow} \rangle$ be two reduction systems having the same object set A.

1.3.5. Definition. $R_1 = \langle A, \underset{1}{\rightarrow} \rangle$ commutes with $R_2 = \langle A, \underset{2}{\rightarrow} \rangle$ (denoted by $COM(R_1, R_2)$) iff R_1 and R_2 satisfy the diagram in Figure 1.2.

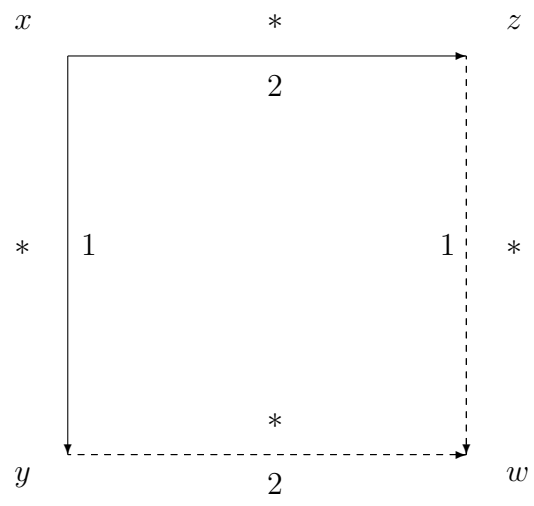


Figure 1.2

Note that R has the Church-Rosser property iff R is self-commuting, i.e., R

commutes with itself. Hindley [34] and Rosen [81] discovered the following useful propositions.

1.3.6. Proposition (Commutative Union Theorem). Let $R_i = \langle A, \underset{i}{\rightarrow} \rangle$ $(i \in I)$ be reduction systems. Let R_i commute with R_j for all $i, j \in I$. Then $\underset{i \in I}{\cup} R_i$ has the Church-Rosser property, where $\underset{i \in I}{\cup} R_i = \langle A, \underset{i \in I}{\cup} \underset{i}{\rightarrow} \rangle$.

1.3.7. Proposition (Commutativity Lemma). Let R_1 and R_2 satisfy the diagram in Figure 1.3. Then R_1 commutes with R_2.

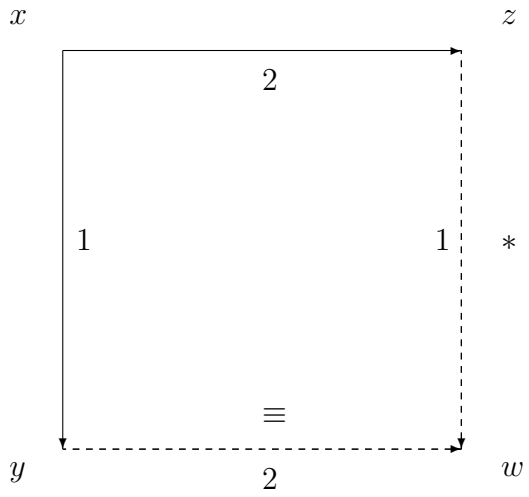


Figure 1.3

1.4. Term Rewriting Systems

A term rewriting system is a reduction system having a term set as an object set A. This section explains the basic notions and properties of term rewriting systems, according to Huet and Oppen [39], Klop [58].

Let F be an enumerable set of function symbols denoted by $f, g, h, \cdots$, and let V be an enumerable set of variable symbols denoted by $x, y, z, \cdots$ where $F \cap V = \phi$. By $T(F, V)$, we denote the set of terms constructed from F and V. An arity function ρ is a mapping from F to natural numbers $\mathbf{N}$, and if $\rho(f) = n$ then f is called an n-ary function symbol. In particular, a 0-ary function symbol is called a constant.

1.4.1. Definition. The set $T(F, V)$ of terms on a function symbol set F is inductively defined as follows:

(i) $x \in T(F, V)$ if $x \in V$,

(ii) $f \in T(F, V)$ if $f \in F$ and $\rho(f) = 0$,

(iii) $f(M_1, \ldots, M_n) \in T(F, V)$ if $f \in F, \rho(f) = n > 0$, and $M_1, \ldots, M_n \in T(F, V)$.

Note. Often we use the infix notation $M_1 f M_2$ instead of $f(M_1, M_2)$ when the arity of f is 2.

Terms containing no variable are called ground terms, and $T(F)$ is the set of ground terms. If every variable in a term occurs only once, then the term is called linear. We use the symbols $M, N, P, \cdots$ for terms and the symbols $T, T', T'', \cdots$ for term sets.

Consider an extra constant $\Box$ called a hole and the set $T(F \cup \{\Box\}, V)$. Then $C \in T(F \cup \{\Box\}, V)$ is called a context on F. We use the notation $C[\ , \ldots, \ ]$ for the context containing n holes ($n \geq 0$), and if $N_1, \ldots, N_n \in T(F, V)$, then $C[N_1, \ldots, N_n]$ denotes the result of placing $N_1, \ldots, N_n$ in the holes of $C[\ , \ldots, \ ]$ from left to right. In particular, $C[\ ]$ denotes a context containing precisely one hole.

N is called a subterm of M if $M \equiv C[N]$. Let N be a subterm occurrence of M; then, we write $N \subseteq M$, and if $N \not\equiv M$, then we write $N \subset M$. If N_1 and N_2 are subterm occurrences of M having no common symbol occurrences (i.e., $M \equiv C[N_1, N_2]$), then N_1, N_2 are called disjoint (denoted by $N_1 \perp N_2$).

1.4.2. Example. Let $F = \{f, g, c\}$ where the arities are $\rho(f) = 2$, $\rho(g) = 1$, and $\rho(c) = 0$. Then $f(x, g(x))$ is a (non-linear) term, $f(x, g(y))$ is a linear term, $f(c, g(c))$ is a ground term, $g(c)$ and $f(x, y)$ are subterms of $f(g(c), f(x, y))$ and $g(c) \perp f(x, y)$. $\square$

1.4.3. Definition. A substitution θ is a mapping from a term set T to T such that;

(i) $\theta(f) = f$ if f is constant,

(ii) $\theta(f(M_1, \ldots, M_n)) \equiv f(\theta(M_1), \ldots, \theta(M_n))$ if $f(M_1, \ldots, M_n) \in T$.

Thus, for a term M, $\theta(M)$ is determined by its values on the variable symbols occurring in M. Following common usage, we write this as $M\theta$ instead of $\theta(M)$.

A rewriting rule is a pair $\langle M_l, M_r \rangle$ of terms in $T(F, V)$ such that $M_l \notin V$ and any variable in M_r also occurs in M_l. The notation $\triangleright$ denotes a set of rewriting rules and we write $M_l \triangleright M_r$ for $\langle M_l, M_r \rangle \in \triangleright$. A $\rightarrow$redex, or redex, is a term $M_l\theta$, where $M_l \triangleright M_r$, and in this case $M_r\theta$ is called a $\rightarrow$contractum, of $M_l\theta$. The set $\triangleright$ of rewriting rules defines a reduction relation $\rightarrow$ on T as follows:

$$M \rightarrow N \text{ iff } M \equiv C[M_l\theta],\ N \equiv C[M_r\theta], \text{ and } M_l \triangleright M_r \text{ for some } M_l, M_r, C[\ ], \text{ and } \theta.$$

When we want to specify the redex occurrence $A \equiv M_l\theta$ of M in this reduction, write $M \xrightarrow{A} N$.

1.4.4. Definition. A term rewriting system R on T is a reduction system $R = \langle T, \rightarrow \rangle$ such that the reduction relation $\rightarrow$ is defined by a set $\triangleright$ of rewriting rules on T. If R has $M_l \triangleright M_r$, then we write $M_l \triangleright M_r \in R$.

We say that R is left-linear (or linear) iff for any $M_l \triangleright M_r \in R$, M_l is linear. R

is called non-left-linear (or non-linear) if R is not left-linear.

1.4.5. Example. Consider $F = \{+, s, 0\}$ where arities are 2, 1, 0 respectively. Let R be the (left-linear) term rewriting system on $T(F, V)$ with the following rewriting rules:

$$R \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y). \end{cases}$$

Then R computes the addition of natural numbers. For instance, $s(0) + s(s(0)) \xrightarrow{*} s(s(s(0)))$ since we have the following reduction:

$$s(0) + s(s(0)) \to s(s(0) + s(0)) \to s(s(s(0) + 0)) \to s(s(s(0))). \ \square$$

1.4.6. Example (Combinatory Logic). Consider $F = \{\bullet, S, K\}$ where arities are 2, 0, 0 respectively. Then combinatory logic (CL) is the (left-linear) term rewriting system on $T(F, V)$ having the following rewriting rules:

$$CL \quad \begin{cases} ((S \bullet x) \bullet y) \bullet z \triangleright (x \bullet z) \bullet (y \bullet z) \\ (K \bullet x) \bullet y \triangleright x. \end{cases}$$

The term rewriting system CL has 'universal' computational power, that is, every (partial) recursive function on the natural numbers can be expressed in CL: See [2]. $\square$

Let $M \triangleright N$ and $P \triangleright Q$ be two rules in R. We assume that we have renamed the variables appropriately, so that M and P share no variables. Assume $S \notin V$ is a subterm occurrence in M, i.e., $M \equiv C[S]$, such that S and P are unifiable, i.e., $S\theta \equiv P\theta$, with a minimal unifier θ [37, 59] . Since $M\theta \equiv C[S]\theta \equiv C\theta[P\theta]$,

two reductions starting with $M\theta$, i.e., $M\theta \to C\theta[Q\theta] \equiv C[Q]\theta$ and $M\theta \to N\theta$, can be obtained by using $P \triangleright Q$ and $M \triangleright N$. Then we say that $P \triangleright Q$ and $M \triangleright N$ are overlapping, and that the pair $\langle C[Q]\theta, N\theta\rangle$ of terms is critical in R [37, 39] . We may choose $M \triangleright N$ and $P \triangleright Q$ to be the same rule, but in this case we shall not consider the case $S \equiv M$, which gives the trivial pair $\langle N, N\rangle$. If R has no critical pair, then we say that R is non-overlapping [37, 39, 59]. Note that the term rewriting systems in Examples 1.4.5 and 1.4.6 both are non-overlapping.

1.4.7. Example. Consider $F = \{*, e\}$ where arities are 2, 0 respectively. Let R be the (left-linear) term rewriting system on $T(F, V)$ with the following rewriting rules:

$$R \quad \begin{cases} (1) \quad e * x \triangleright x \\ (2) \quad (x * y) * z \triangleright x * (y * z). \end{cases}$$

Then (1) and (2) are overlapping at $(e * y) * z$, and (2) and (2) itself are overlapping at $((x * y) * z) * w$. Thus we have the critical pairs $\langle y * z, e * (y * z)\rangle$ and $\langle (x * (y * z)) * w, (x * y) * (z * w)\rangle$ respectively. □

The following sufficient conditions for the Church-Rosser property are well known [37, 39, 59, 81].

1.4.8. Proposition (Knuth-Bendix). Let R be strongly normalizing. Then R has the Church-Rosser property iff P and Q have the same normal form for any critical pair $\langle P, Q\rangle$ in R.

1.4.9. Proposition (Rosen). Let R be left-linear and non-overlapping. Then R has the Church-Rosser property.

Rosen's proposition for the Church-Rosser property of left-linear term rewriting systems is a special case of Huet's condition, stated here. We need a definition first:

1.4.10. Definition. For a term rewriting system R, the parallel reduction $\twoheadrightarrow$ for disjoint redex occurrences is defined as follows. Let $M \equiv C[A_1, \cdots, A_m]$ and let $A_i \xrightarrow{A_i} B_i \ \ (i = 1, \cdots, m)$. Let $N \equiv C[B_1, \cdots, B_m]$. Then we write $M \twoheadrightarrow N$ or $M \overset{A_1,\cdots,A_m}{\twoheadrightarrow} N$.

1.4.11. Proposition (Huet). Let R be left-linear. If $P \twoheadrightarrow Q$ for every critical pair $\langle P, Q\rangle$ in R, then R has the Church-Rosser property.

By applying Propositions 1.4.8 to Examples 1.4.5 and 1.4.7, and Proposition 1.4.9 (or 1.4.11) to Example 1.4.6, we can easily show the Church-Rosser property of the term rewriting systems in the above examples.

The following example shows that *strongly normalizing* (i.e., *terminating*) in Proposition 1.4.8 and *left-linear* in Propositions 1.4.9 and 1.4.11 are necessary.

1.4.12. Example. Consider $F = \{f, g, a, b\}$ where arities are 2, 1, 0, 0 respectively. Let R be the term rewriting system on $T(F, V)$ with the following rewriting rules:

$$R \quad \begin{cases} f(x, x) \triangleright a \\ g(x) \triangleright f(x, g(x)) \\ b \triangleright g(b) \end{cases}$$

Then R is not Church-Rosser, since the term b has two reductions $b \xrightarrow{*} a$ and $b \xrightarrow{*} g(a)$ but $g(a)$ cannot be reduced into the normal form a. Note that R is non-overlapping, but has the non-left-linear rule $f(x, x) \triangleright a$ and an infinite reduction $b \to g(b) \to$

$g(g(b)) \to \cdots$. $\square$

Remark. For term rewriting systems, it is undecidable whether the Church-Rosser property holds, whether the termination property (i.e., strong normalizing) holds [22, 39, 58]. For particular term rewriting systems, it may be undecidable whether two terms are convertible (i.e., they can be connected by $=$), whether a term has a normal form, whether a term has an infinite reduction; for instance, combinatory logic (CL) is a term rewriting system where all these properties are undecidable [2]. For the decidability of *ground* term rewriting systems, see [74, 90].

2. Direct Sum of Term Rewriting Systems

The direct sum of two term rewriting systems is the union of systems having disjoint sets of function symbols. In this chapter it is proven that the Church-Rosser property is modular: if two term rewriting systems both have the Church-Rosser property respectively then the direct sum of these systems also has this property. Moreover, it is shown that the left-linear and completeness property is modular, but the termination property is not.

2.1. Introduction

An important concern in building algebraic specifications is their hierarchical or modular structure. The same holds for term rewriting systems [22, 39, 58] which can be viewed as implementations of equational algebraic specifications. Specifically, it is of obvious interest to determine which properties of term rewriting systems have a *modular* character, where we call a property *modular* if its validity for a term rewriting system, hierarchically composed of some smaller term rewriting systems, can be inferred from the validity of that property for the constituent term rewriting systems. Naturally, the first step in such an investigation considers the most basic properties of term rewriting systems: Church-Rosser, termination, and similar fundamental properties as well as combinations thereof.

As to the modular structure of term rewriting systems, it is again natural to consider as a start the most simple way that term rewriting systems can be combined to form a larger term rewriting system: namely, as a disjoint sum. This means that the alphabets of the term rewriting systems to be combined are disjoint, and that the rewriting rules of the sum term rewriting system are the rules of the summand term rewriting systems together. (Without the disjointness requirement the situation is even more complicated - see Chapter 3 for some results in this direction.) A disjoint union of two term rewriting systems R_1 and R_2 is called a direct sum, notation $R_1 \oplus R_2$.

In this chapter, we consider properties of the direct sum system $R_1 \oplus R_2$ obtained from two term rewriting systems R_1 and R_2. The main result of this chapter, which is proven in Section 2.2, is that the Church-Rosser property is modular: term rewriting systems R_1 and R_2 are Church-Rosser iff the direct sum $R_1 \oplus R_2$ is so. Section 2.3 shows that the left-linear and completeness (i.e., Church-Rosser and termination) property is also modular, but the termination property is not.

2.2. The Church-Rosser Property for the Direct Sum of Term Rewriting Systems

The first study on the direct sum system was conducted by Klop in [56] in order to consider the Church-Rosser property for combinatory reduction systems having nonlinear rewriting rules, which contain term rewriting systems as a special case. He showed that if R_1 is a regular, i.e., linear (i.e., left-linear) and nonoverlapping, system and R_2 consists of the single nonlinear rule $D(x, x) \triangleright x$, then the direct sum $R_1 \oplus R_2$ has the Church-Rosser property. He also showed in the same manner that if R_2 consists of the nonlinear rules

$$R_2 \quad \begin{cases} \text{if}(T, x, y) \triangleright x \\ \text{if}(F, x, y) \triangleright y \\ \text{if}(z, x, x) \triangleright x \end{cases}$$

then the direct sum $R_1 \oplus R_2$ also has the Church-Rosser property. This result gave a positive answer for an open problem suggested by O'Donnell [72].

Klop's work was done on combinatory reduction systems having the following restrictions: R_1 is a regular (i.e., linear and nonoverlapping) system, and R_2 is a nonlinear system having specific rules such as $D(x, x) \triangleright x$. In particular, the restriction on R_1 plays an essential role in his proof of the Church-Rosser property of $R_1 \oplus R_2$; hence his result cannot be applied to combinatory reduction systems (and term rewriting systems) without this restriction.

From Klop's work, we consider the conjecture that these restrictions can be completely removed from R_1 and R_2 in the framework of term rewriting systems [37], i.e., the direct sum of term rewriting systems R_1 and R_2, independent of their properties such as linear or nonoverlapping, always preserves their Church-Rosser property. In this section we shall prove this conjecture: For any two term rewriting systems R_1 and R_2, R_1 and R_2 have the Church-Rosser property iff $R_1 \oplus R_2$ has this property.

Note. In this section there are no limitations on term rewriting systems, thus, they may have nonlinear or overlapping rewriting rules [37, 56].

2.2.1. Direct Sum Systems

Let F_1 and F_2 be disjoint sets of function symbols (i.e., $F_1 \cap F_2 = \phi$), then term rewriting systems R_1 on $T(F_1, V)$ and R_2 on $T(F_2, V)$ are called disjoint. Consider disjoint systems R_1 and R_2 having sets $\underset{1}{\triangleright}$ and $\underset{2}{\triangleright}$ of rewriting rules, respectively,

then the direct sum system $R_1 \oplus R_2$ is the term rewriting system on $T(F_1 \cup F_2, V)$ having the set $\underset{1}{\triangleright} \cup \underset{2}{\triangleright}$ of rewriting rules. If R_1 and R_2 are term rewriting systems not satisfying the disjoint requirement for function symbols, then we take isomorphic copies R_1' and R_2' by replacing each function symbol f of F_i by f^i $(i = 1, 2)$, and use $R_1' \oplus R_2'$ instead of $R_1 \oplus R_2$. For this reason, considering the direct sum $R_1 \oplus R_2$, we may assume that R_1 and R_2 are always disjoint, i.e., $F_1 \cap F_2 = \phi$.

Note. The above direct sum is different from Klop's [56]: The direct sum of combinatory reduction systems (in which terms are written in *combinator notation*) is defined as the union of two systems with disjoint constant symbols, but with the same application function symbol. Klop pointed out that his direct sum does not preserve the Church-Rosser property.

It is trivial that if $CR(R_1 \oplus R_2)$ then $CR(R_1)$ and $CR(R_2)$. Hence, in the following subsections we shall prove $CR(R_1 \oplus R_2)$, assuming that $CR(R_1)$ and $CR(R_2)$ where $R_1 = \langle T(F_1, V), \underset{1}{\rightarrow} \rangle$, $R_2 = \langle T(F_2, V), \underset{2}{\rightarrow} \rangle$, and $R_1 \oplus R_2 = \langle T(F_1 \cup F_2, V), \rightarrow \rangle$. Note that from here on the notation $\rightarrow$ represents the reduction relation on $R_1 \oplus R_2$.

2.2.1.1. Definition. A *root* of a term $M \in T(F_1 \cup F_2, V)$ is defined by

$$root(M) = \begin{cases} f & \text{if } M \equiv f(M_1, \ldots, M_n), \\ M & \text{if } M \text{ is a constant or a variable.} \end{cases}$$

2.2.1.2. Definition. Let $M \equiv C[B_1, \ldots, B_n] \in T(F_1 \cup F_2, V)$ and $C \not\equiv \Box$. Then write $M \equiv C[\![B_1, \ldots, B_n]\!]$ if $C[\ , \ldots, \ ]$ is a context on F_a and $\forall i, root(B_i) \in F_b$ $(a, b \in \{1, 2\}$and $a \neq b)$. Then the set $S(M)$ of the special subterms of $M \in T(F_1 \cup F_2, V)$ is inductively defined as follows:

$$S(M) = \begin{cases} \{M\} & \text{if } M \in T(F_a, V) \ \ (a = 1 \text{ or } 2), \\ \bigcup_i S(B_i) \cup \{M\} & \text{if } M \equiv C[\![B_1, \ldots, B_n]\!] \ (n > 0). \end{cases}$$

2.2.1.3. Definition. For a term $M \in T(F_1 \cup F_2, V)$, the rank of layers of

contexts on F_1 and F_2 in M is inductively defined as follows:

$$rank(M) = \begin{cases} 1 & \text{if } M \in T(F_a, V) \ \ (a = 1 \text{ or } 2), \\ max_i\{rank(B_i)\} + 1 & \text{if } M \equiv C[\![B_1, \ldots, B_n]\!] \ (n > 0). \end{cases}$$

2.2.1.4. Example. Let a rewriting rule of R_1 be $f(x) \triangleright f(f(x))$, and let a rewriting rule of R_2 be $g(x, x) \triangleright x$, where $F_1 = \{f\}, F_2 = \{g\}, \rho(f) = 1, \rho(g) = 2$. Consider a term $M_0 \equiv g(f(x), g(f(f(g(x, x))), f(x))) \in T(F_1 \cup F_2, V)$. Note that M_0 has a layer structure of contexts on F_1 and F_2 constructed by $g(\Box, g(\Box, \Box))$ on F_2, $f(x), f(f(\Box)), f(x)$ on F_1, and $g(x, x)$ on F_2 from the outside. Then $S(M_0) = \{M_0, f(x), f(f(g(x, x))), g(x, x)\}$, $root(M_0) = g$. We can write
$M_0 \equiv C[\![f(x), f(f(g(x, x))), f(x)]\!]$ where $C[\ ,\ ,\ ] \equiv g(\Box, g(\Box, \Box))$.

$R_1 \oplus R_2$ has the following reduction:

$M_0 \equiv g(f(x), g(f(f(g(x, x))), f(x)))$

$\rightarrow\ M_1 \equiv g(f(x), g(f(f(x)), f(x)))$

$\rightarrow\ M_2 \equiv g(f(x), g(f(f(x)), f(f(x))))$

$\rightarrow\ M_3 \equiv g(f(x), f(f(x)))$

$\rightarrow\ M_4 \equiv g(f(f(x)), f(f(x)))$

$\rightarrow\ M_5 \equiv f(f(x))$.

Then $rank(M_0) = 3, rank(M_1) = rank(M_2) = rank(M_3) = rank(M_4) = 2, rank(M_5) = 1$.

2.2.1.5. Lemma. If $M \rightarrow N$ then $rank(M) \geq rank(N)$.

Proof. It is easily obtained from the definitions of the direct sum $R_1 \oplus R_2$. □

2.2.2. Preserved Systems

A term $M \in T(F_1 \cup F_2, V)$ has a layer structure of contexts on F_1 and F_2, and this structure is modified through a reduction process in a direct sum system $R_1 \oplus R_2$ on $T(F_1 \cup F_2, V)$. If a reduction $M \to N$ results in the disappearance of some layer between two layers in the term M, then, by putting the two layers together, a new layer structure appears in the term N. If no middle layer in M disappears as a result of any reduction, then we say that the layer structure in M is preserved in the direct sum system. In this subsection we will show that if two term rewriting systems have the Church-Rosser property, then terms with a certain restriction, namely, that their layer structure is preserved under reductions, also have the Church-Rosser property. Using this result, we will prove our conjecture in Subsection 2.2.3.

The set of terms reduced from a term M by a reduction relation $\to$ is denoted by $G_{\to}(M) = \{N |\ M \xrightarrow{*} N\}$.

2.2.2.1 Definition. A term M is root preserved (denoted by $r\text{-}Pre(M)$) iff $root(M) \in F_a \Rightarrow \forall N \in G_{\to}(M), root(N) \in F_a$, where $a \in \{1, 2\}$.

Now we formalize the concept of preserved layer structure.

2.2.2.2 Definition. A term $M \equiv C[\![B_1, \ldots, B_n]\!]$ $(n > 0)$ is preserved iff M satisfies two conditions;

(1) $r\text{-}Pre(M)$,

(2) $\forall i, B_i$ is preserved.

We write $Pre(M)$ when M is preserved. Note that, by the definition, if $Pre(M)$, then $\forall N \in G_{\to}(M), Pre(N)$.

Let $M \xrightarrow{A} N$ and $M \equiv C[\![B_1, \ldots, B_n]\!]$. If the redex occurrence A occurs in some B_j, then we write $M \underset{i}{\to} N$; otherwise $M \underset{o}{\to} N$. $\underset{i}{\to}$ and $\underset{o}{\to}$ are called an inner and an outer reduction, respectively.

2.2.2.3 Lemma. Let $Pre(M)$ and $M \equiv C[\![B_1, \ldots, B_n]\!]$. Then,

(1) $M \underset{i}{\rightarrow} N \Rightarrow N \equiv C[\![C_1, \ldots, C_n]\!]$ where $\forall i, B_i \overset{\equiv}{\rightarrow} C_i$;

(2) $M \underset{o}{\rightarrow} N \Rightarrow N \equiv C'[\![B_{i_1}, \ldots, B_{i_p}]\!]$ $(1 \leq i_j \leq n)$, where $C[\ , \ldots, \ ]$ and $C'[\ , \ldots, \ ]$ are contexts on the same set F_a $(a = 1 \text{ or } 2)$.

Proof. It is immediately proved from $Pre(M)$ and the definition of $\underset{i}{\rightarrow}$, $\underset{o}{\rightarrow}$. □

We consider the term sequences; $\alpha = \langle A_1, \ldots, A_n \rangle$ and $\beta = \langle B_1, \ldots, B_n \rangle$, where $A_i, B_i \in T$. Then, we write $\alpha \propto \beta$ iff $\forall i, j[A_i \equiv A_j \Rightarrow B_i \equiv B_j]$. We define $\alpha \overset{*}{\rightarrow} \beta$ by $\forall i, A_i \overset{*}{\rightarrow} B_i$.

We extend the above notations to terms. Let $M \equiv C[\![A_1, \ldots, A_n]\!]$, $N \equiv C[\![B_1, \ldots, B_n]\!]$, $\alpha = \langle A_1, \ldots, A_n \rangle$, $\beta = \langle B_1, \ldots, B_n \rangle$. Then write $M \propto N$ if $\alpha \propto \beta$.

We use the relation $\propto$ to deal with nonlinear rewriting rules. For example, let the reduction $f(A_1, A_2, A_3, A_4) \overset{*}{\rightarrow} g(A_1)$ be obtained by using the nonlinear rule $f(x, x, y, y) \triangleright g(x)$. Then, we can obtain the reduction $f(B_1, B_2, B_3, B_4) \overset{*}{\rightarrow} g(B_1)$ by the same rule if $\langle A_1, A_2, A_3, A_4 \rangle \propto \langle B_1, B_2, B_3, B_4 \rangle$. This leads us to the following lemma.

2.2.2.4. Lemma. Let $Pre(M)$, $M \propto N$. If $M \underset{o}{\rightarrow} M'$, then $\exists N', N \underset{o}{\rightarrow} N' \wedge M' \propto N'$.

Proof. Let $M \equiv C[\![A_1, \ldots, A_n]\!]$, $N \equiv C[\![B_1, \ldots, B_n]\!]$. Then the left side of the rewriting rule used in $M \underset{o}{\rightarrow} M'$ occurs in context $C[\ , \ldots, \ ]$. Since $M \propto N$ we can apply this rule to N in the same way, and obtain $N \underset{o}{\rightarrow} N'$. By Lemma 2.2.2.3(2), it is clear that $M' \propto N'$. □

2.2.2.5. Lemma. Let $Pre(M)$, $M \underset{o}{\rightarrow} P$, $M \overset{*}{\underset{i}{\rightarrow}} N$, $M \propto N$. Then there is a term Q satisfying the diagram in Figure 2.1, that is,

$\forall M, N, P \in T[M \overset{*}{\underset{i}{\rightarrow}} N \wedge M \underset{o}{\rightarrow} P \wedge M \propto N \Rightarrow \exists Q \in T, N \underset{o}{\rightarrow} Q \wedge P \overset{*}{\underset{i}{\rightarrow}} Q \wedge P \propto Q]$.

Proof. By Lemma 2.2.2.4 we obtain a term Q such that $P \propto Q$ and $N \underset{o}{\rightarrow} Q$. Using $M \underset{o}{\rightarrow} P$, $M \underset{i}{\overset{*}{\rightarrow}} N$ and Lemma 2.2.2.3(1), (2), we obtain $P \underset{i}{\overset{*}{\rightarrow}} Q$. □

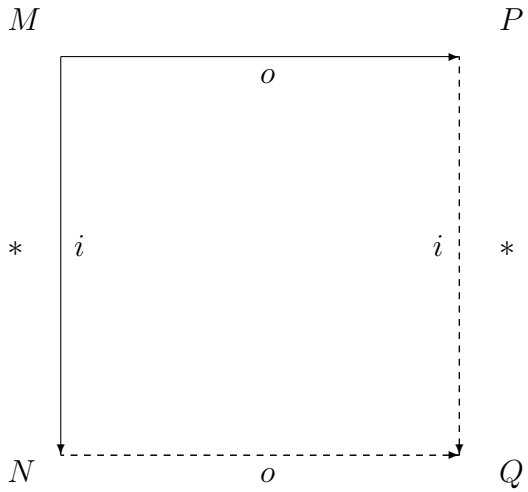


Figure 2.1

2.2.2.6. Lemma. Let $Pre(M)$, $M \underset{i}{\overset{*}{\rightarrow}} N$, $M \underset{o}{\overset{*}{\rightarrow}} P$, $M \propto N$. Then we can obtain a term Q satisfying Figure 2.2.

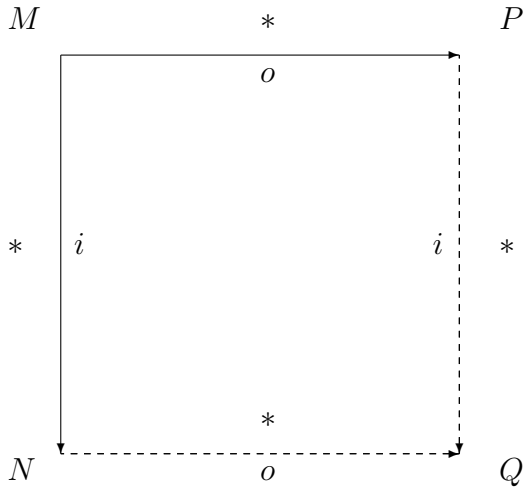


Figure 2.2

Proof. Using lemma 2.2.2.5, the diagram in Figure 2.3 can be made. □

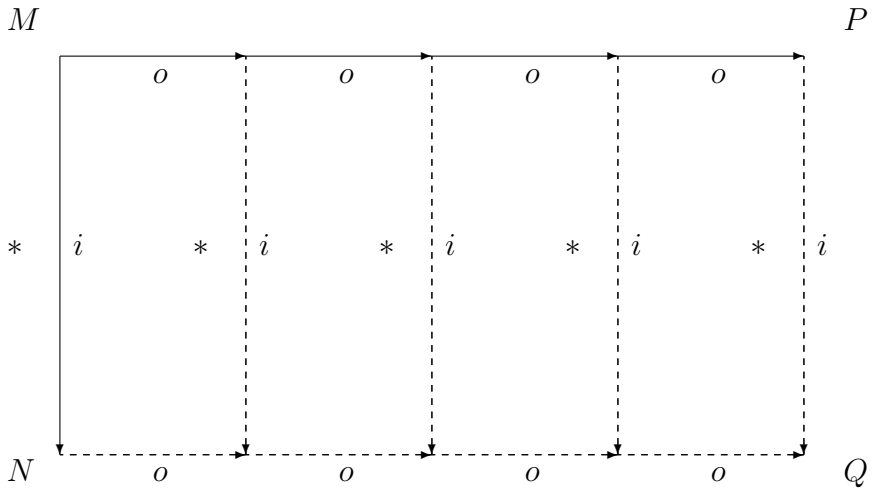


Figure 2.3

We define the local Church-Rosser property at a term M.

2.2.2.7. Definition. Let $R = \langle T, \rightarrow \rangle$ be a reduction system and let $M \in T$. Then M is Church-Rosser for $\rightarrow$ (denoted by $CR_{\rightarrow}(M)$ or $CR(M)$) iff

$\forall N, P \in T[M \xrightarrow{*} N \wedge M \xrightarrow{*} P \Rightarrow \exists Q \in T, N \xrightarrow{*} Q \wedge P \xrightarrow{*} Q]$.

Note that $\forall M \in T, CR(M)$ iff $CR(R)$.

We define $M \downarrow N$ by $\exists Q \in T, M \xrightarrow{*} Q \wedge N \xrightarrow{*} Q$.

2.2.2.8. Lemma. Let $\alpha = \langle A_1, \ldots, A_n \rangle$ and $\forall i, CR(A_i)$. Then

$\exists \beta = \langle B_1, \ \ldots, B_n \rangle [\alpha \xrightarrow{*} \beta \wedge \forall i, j[A_i \downarrow A_j \Rightarrow B_i \equiv B_j]]$.

Proof. Using $CR(A_k)$, it can be shown that $A_i \downarrow A_k \wedge A_k \downarrow A_j \Rightarrow A_i \downarrow A_j$. Hence $\downarrow$ is an equivalence relation and it partitions $\{A_1, \ldots, A_n\}$ in the equivalence class $C_1, \ldots, C_m$. Using the Church-Rosser property for each A_i, we can take a term B_p for each equivalence class $C_p = \{A_{p_1}, \ldots, A_{p_q}\}$ as the diagram in Figure 2.4. Take $B_{p_1} \equiv \ldots \equiv B_{p_q} \equiv B_p$. $\square$

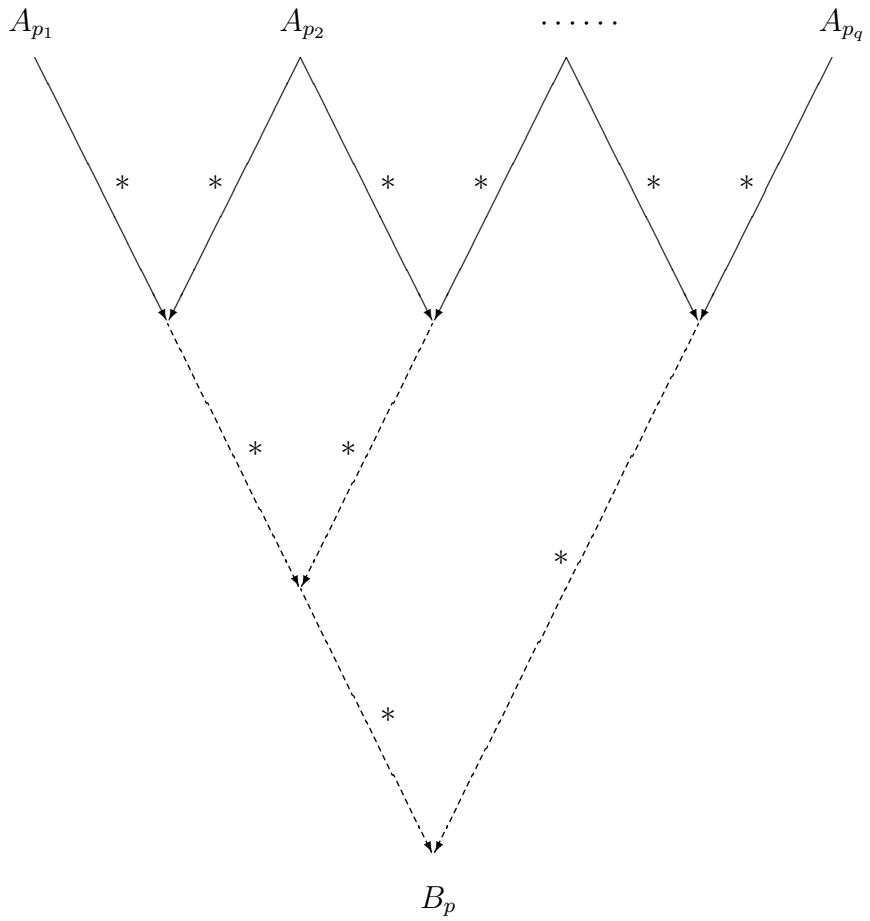


Figure 2.4

2.2.2.9. Lemma. Let $\alpha = \langle A_1, \ldots, A_n \rangle \xrightarrow{*} \beta = \langle B_1, \ldots, B_n \rangle$ and $\forall i, CR(A_i)$. Then $A_i \downarrow A_j$ iff $B_i \downarrow B_j$.

Proof. By the Church-Rosser property for each A_i, it is obvious. □

2.2.2.10. Lemma. Let $\alpha = \langle A_1, \ldots, A_n \rangle$, $\forall i, CR(A_i)$, and $\alpha \xrightarrow{*} \beta$, $\alpha \xrightarrow{*} \gamma$. Then we can obtain δ satisfying Figure 2.5, where $\beta \propto \delta$ and $\gamma \propto \delta$.

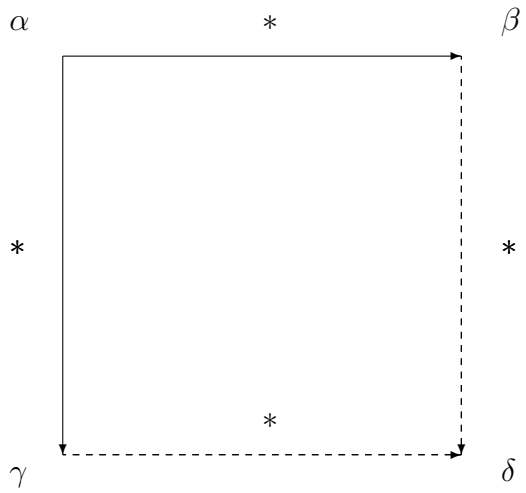


Figure 2.5

Proof. Let $\beta = \langle B_1, \ldots, B_n \rangle$, $\gamma = \langle C_1, \ldots, C_n \rangle$. By $\forall i, CR(A_i)$, we have a term $\delta' = \langle D'_1, \ldots, D'_n \rangle$ such that $\beta \xrightarrow{*} \delta'$ and $\gamma \xrightarrow{*} \delta'$. Using Lemma 2.2.2.8 for δ', we obtain $\delta = \langle D_1, \ldots, D_n \rangle$ such that $\delta' \xrightarrow{*} \delta$ and $D'_i \downarrow D'_j \Rightarrow D_i \equiv D_j$. Then, by Lemma 2.2.2.9, $A_i \downarrow A_j \iff D'_i \downarrow D'_j$, hence $A_i \downarrow A_j \Rightarrow D_i \equiv D_j$. Next we show $\beta \propto \delta$. If $B_i \equiv B_j$, then $A_i \downarrow A_i$, and, thus $D_i \equiv D_j$, hence $\beta \propto \delta$. Similarly we can prove $\gamma \propto \delta$. □

2.2.2.11. Lemma. Let $M \equiv C[\![A_1, \ldots, A_n]\!]$, $Pre(M)$, $\forall i, CR(A_i)$. Then we have the diagram in Figure 2.6, where $N \propto Q$, $P \propto Q$.

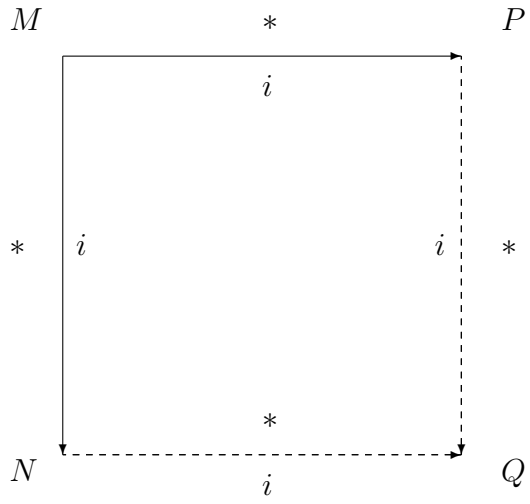


Figure 2.6

Proof. Since $Pre(M)$, we obtain $N \equiv C[\![B_1, \ldots, B_n]\!], P \equiv C[\![C_1, \ldots, C_n]\!]$, where $\alpha = \langle A_1, \ldots, A_n \rangle \xrightarrow{*} \beta = \langle B_1, \ldots, B_n \rangle$, $\alpha = \langle A_1, \ldots, A_n \rangle \xrightarrow{*} \gamma = \langle C_1, \ldots, C_n \rangle$. Using Lemma 2.2.2.10, we can obtain $\delta = \langle D_1, \ldots, D_n \rangle$ such that $\beta \xrightarrow{*} \delta$, $\gamma \xrightarrow{*} \delta$, $\beta \propto \delta$ and $\gamma \propto \delta$. Therefore, take $Q \equiv C[\![D_1, \ldots, D_n]\!]$. □

2.2.2.12. Lemma. If $Pre(M)$, then $CR_{\underset{o}{\rightarrow}}(M)$, that is, M is Church-Rosser for $\underset{o}{\rightarrow}$ (Figure 2.7).

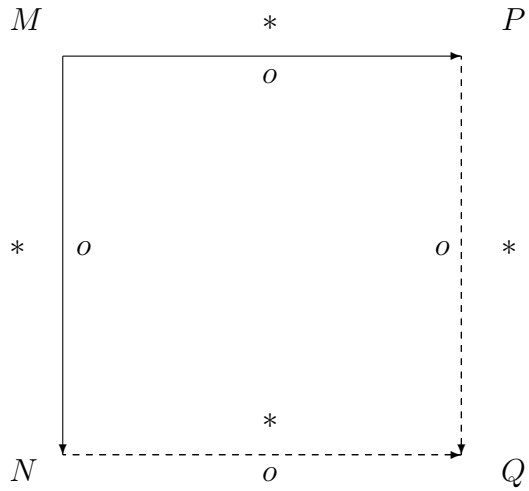


Figure 2.7

Proof. Let $root(M) \in F_a$ $(a = 1 \text{ or } 2)$. Then, since $Pre(M)$, the outermost part of any term in $G_{\rightarrow}(M)$ is always a context on F_a. Thus $\underset{o}{\rightarrow}$ is determined by only R_a. Hence Church-Rosser for $\underset{o}{\rightarrow}$ is obvious by $CR(R_a)$. □

2.2.2.13. Theorem. If $Pre(M)$, then $CR(M)$.

Proof. By induction on the rank $rank(M)$ of layers in M. The case $rank(M) = 1$ is trivial since $M \in T(F_a, V)$ and $CR(R_a)$ $(a = 1 \text{ or } 2)$; therefore, suppose $rank(M) = n > 1$, $M \equiv C[\![A_1, \ldots, A_m]\!]$.

Claim: We obtain the diagram in Figure 2.8.

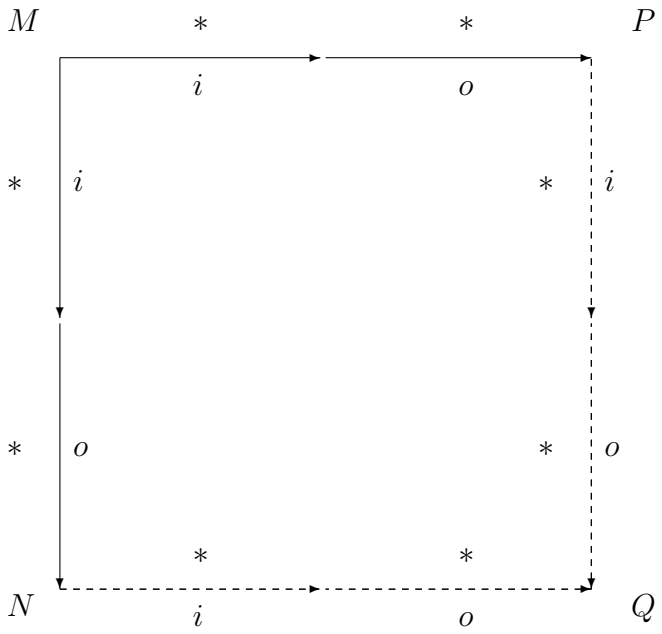


Figure 2.8

Proof of the claim. By the induction hypothesis, we obtain $\forall i, CR(A_i)$. Using Lemmas 2.2.2.11, 2.2.2.6 and 2.2.2.12 for (1), (2) and (3), respectively, we can obtain the diagram in Figure 2.9, where $M' \propto Q'$ and $M'' \propto Q'$.

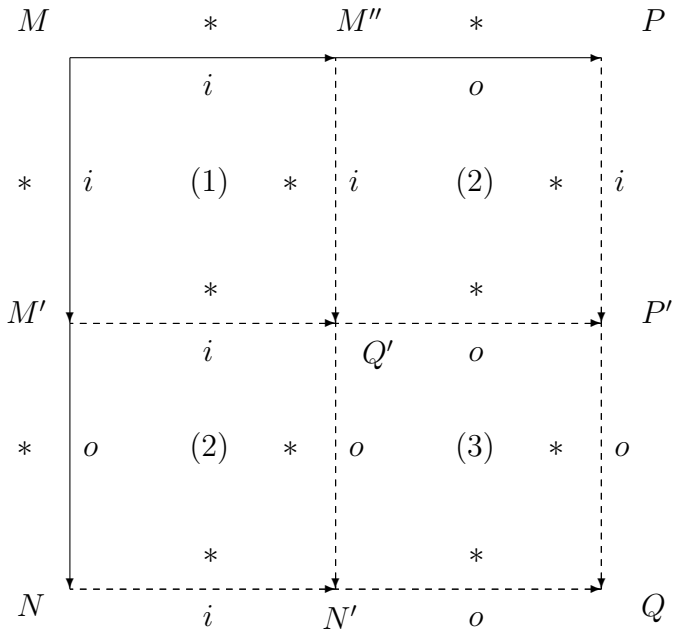


Figure 2.9

Now we will show $CR(M)$. Note that any reduction $M \xrightarrow{*} M'$ takes the form of $M \underset{i}{\xrightarrow{*}} \underset{o}{\xrightarrow{*}} M_1 \underset{i}{\xrightarrow{*}} \underset{o}{\xrightarrow{*}} M_2 \underset{i}{\xrightarrow{*}} \underset{o}{\xrightarrow{*}} \cdots \underset{i}{\xrightarrow{*}} \underset{o}{\xrightarrow{*}} M'$.

Let $M \xrightarrow{*} N$, $M \xrightarrow{*} P$. By splitting $\xrightarrow{*}$ into $\underset{i}{\xrightarrow{*}} \underset{o}{\xrightarrow{*}}$ and using the claim, one can draw the diagram in Figure 2.10. Hence $CR(M)$. $\square$

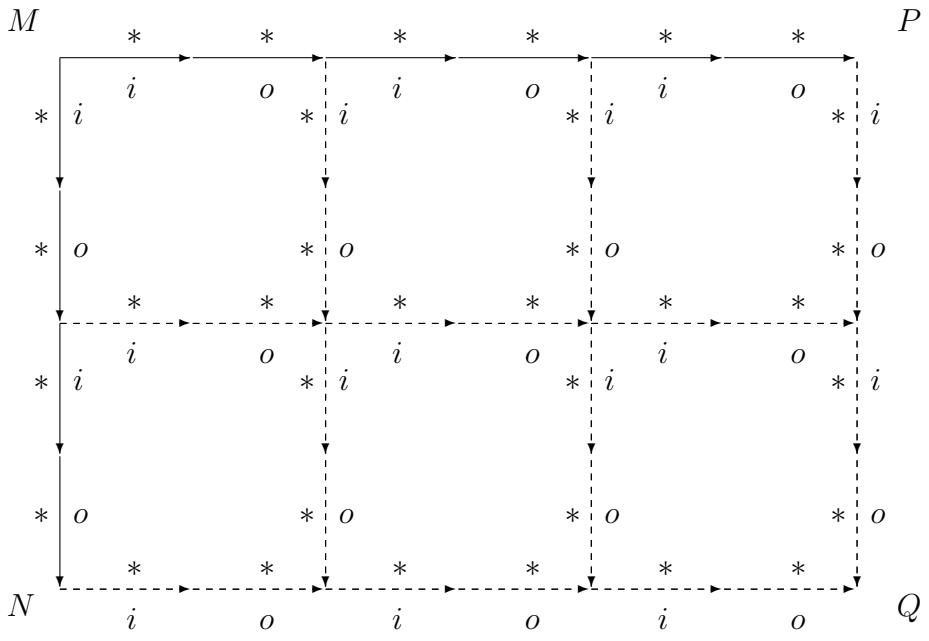


Figure 2.10

Let $M \xrightarrow{A} N$ where A is a redex occurrence. Then write $M \underset{p}{\rightarrow} N$ if A occurs in a preserved subterm of M, otherwise write $M \underset{np}{\rightarrow} N$.

2.2.2.14. Theorem. Let $M \equiv C[\![A_1, \ldots, A_n]\!]$, $\forall i, Pre(A_i)$. Then $CR(M)$.

Proof. If $Pre(M)$, immediate by Theorem 2.2.2.13. Hence, suppose $\neg Pre(M)$. Then one can prove the diagrams (1), (2) and (3) in Figure 2.11, where $M \propto N$ in (1) and $N \propto Q$, $P \propto Q$ in (2), in the same way as for Lemmas 2.2.2.6, 2.2.2.11 and 2.2.2.12, respectively, by replacing $\underset{i}{\rightarrow}$, $\underset{o}{\rightarrow}$ with $\underset{p}{\rightarrow}$, $\underset{np}{\rightarrow}$. Using an analogy to the proof in Theorem 2.2.2.13, first, one can obtain the diagram in Figure 2.12 from the diagrams (1), (2), (3) in Figure 2.11, and secondly, splitting $\xrightarrow{*}$ into $\underset{p}{\xrightarrow{*}} \underset{np}{\xrightarrow{*}}$, one can show $CR(M)$. □

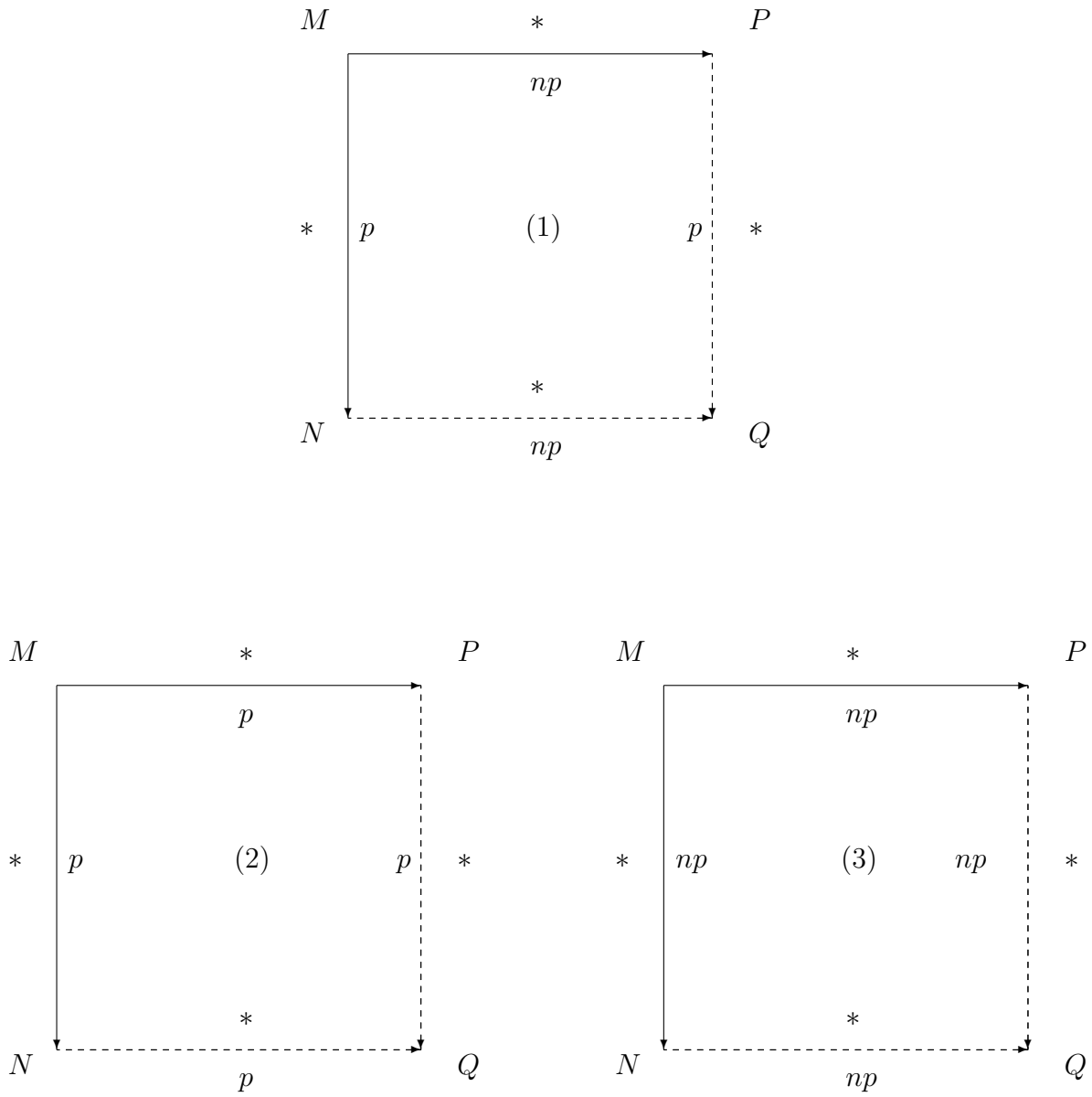


Figure 2.11

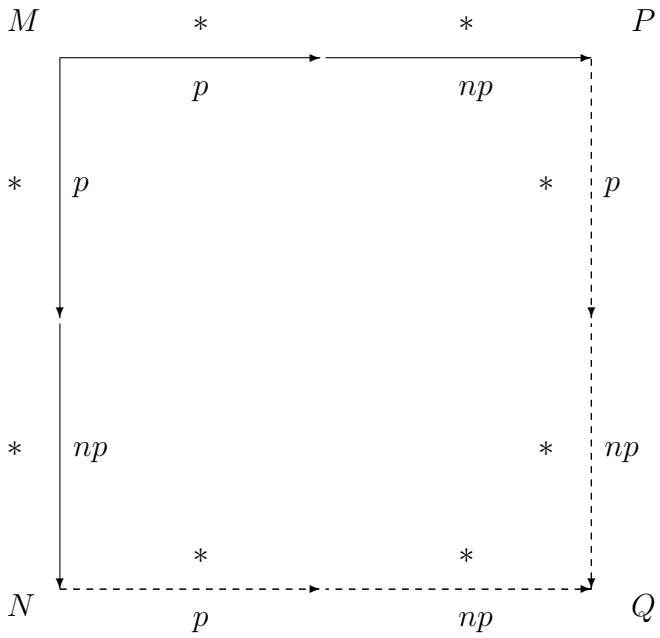


Figure 2.12

Note. Though $\neg Pre(M)$, the above proof is similar to the proof of Theorem 2.2.2.13 in which we assumed $Pre(M)$. This analogy comes from the fact that in Theorem 2.2.2.14 a non-preserved context in a term M only occurs at the outermost part of layer structure. However, if some non-preserved context occurs in the middle part, then one cannot prove $CR(M)$ by the analogous method to Theorem 2.2.2.13. In the next subsection we shall consider this case.

2.2.3. The Church-Rosser Property for the Direct Sum

In this subsection we will show that if $CR(R_1)$ and $CR(R_2)$, then $CR(R_1 \oplus R_2)$. This is done by proving $CR(M)$ for any term M by using parallel deletion reduction which deletes the layers of the non-preserve contexts occurring in M. First we shall introduce the following deletion reduction.

Let a term $M \in T(F_1 \cup F_2, V)$ be not preserved. Then there is a term $N \in S(M)$: $N \equiv \tilde{C}[\![B_1, \ldots, B_n]\!]$, $\neg Pre(N)$, $\forall i, Pre(B_i)$. Since N is not preserved, one has N': $N \xrightarrow{*} N'$, $root(N) \in F_a$, $root(N') \notin F_a$ $(a = 1 \text{ or } 2)$. Then the deletion reduction $\underset{d}{\rightarrow}$ is defined by replacing N occurring in M by N' as follows:

$M \underset{d}{\rightarrow} M' \iff M \equiv C[N],\ M' \equiv C[N']$,

where N and N' are the above terms.

Then we say N is $\underset{d}{\rightarrow}$redex. From this definition, $\underset{d}{\rightarrow} \subseteq \xrightarrow{*}$. Let N_1, N_2 be two different $\underset{d}{\rightarrow}$redex occurrences in M, then it is trivial from the definition that N_1, N_2 are disjoint, that is, $N_1 \perp N_2$. Note that $M \in NF_{\underset{d}{\rightarrow}}$ iff $Pre(M)$.

2.2.3.1. Definition. The maximum depth $d(M)$ of $\underset{d}{\rightarrow}$redex occurrences in M is defined by the following:

$$d(M) = \begin{cases} 0 & \text{if} \quad Pre(M), \\ 1 & \text{if} \quad \neg Pre(M) \text{ and } M \text{ is } \underset{d}{\rightarrow}\text{redex}, \\ max_i\{d(B_i)\} + 1 & \text{if} \quad \neg Pre(M),\ M \text{ is not } \underset{d}{\rightarrow} \text{ redex}, \\ & \quad \text{and } M \equiv C[\![B_1, \ldots, B_n]\!] \ \ (n > 0). \end{cases}$$

2.2.3.2. Lemma. Let $M \equiv C[B_1, \ldots, B_n]$ and $C \in T(F_a \cup \{\Box\}, V)$ $(a = 1 \text{ or } 2)$, then $d(M) \leq max_i\{d(B_i)\} + 1$.

Proof. It is immediately proved from the definition of $d(M)$. □

2.2.3.3. Lemma. If $M \rightarrow N$ then $d(M) \geq d(N)$.

Proof. We will prove the lemma by induction on $d(M)$. The case $d(M) \leq 1$ is trivial from the definition. Assume the lemma for $d(M) < k$ $(k > 1)$; then we show the case $d(M) = k$. Let $M \equiv C[\![B_1, \ldots, B_n]\!]$ $(n > 0)$ and $M \xrightarrow{A} N$.

Case 1. $\exists k, A \subseteq B_k$.

Then $N \equiv C[B_1, \ldots, B_{k-1}, B'_k, B_{k+1}, \ldots, B_n]$ where $B_k \xrightarrow{A} B'_k$. We can obtain $d(B_k) \geq d(B'_k)$ by using the induction hypothesis. Hence by Lemma 2.2.3.2,

$$\begin{aligned} d(M) &= max_i\{d(B_i)\} + 1 \\ &\geq max\{d(B_1), \ldots, d(B_{k-1}), d(B'_k), d(B_{k+1}), \ldots, d(B_n)\} + 1 \\ &\geq d(N). \end{aligned}$$

Case 2. Not Case 1.

Then $N \equiv C'[B_{i_1}, \ldots, B_{i_s}]$ where $1 \leq i_j \leq n$ and $C' \in T(F_a \cup \{\Box\}, V)$ ($a = 1$ or 2). If $s = 0$ then it is clear from $d(N) = 1$ or 0 that $d(M) \geq d(N)$. If $s > 0$ then

$$\begin{aligned} d(M) &= max_i\{d(B_i)\} + 1 \\ &\geq max_j\{d(B_{i_j})\} + 1 \\ &\geq d(N) \end{aligned}$$

for both $C' \equiv \Box$ and $C' \not\equiv \Box$. $\Box$

Let $N_1, \ldots, N_n$ be all the $\underset{d}{\rightarrow}$ redex occurrences in M having depth $d(M)$. Note that $N_i \perp N_j$ $(i \neq j)$. Then the parallel deletion reduction $\underset{pd}{\rightarrow}$ is defined by replacing each $\underset{d}{\rightarrow}$ redex occurrence N_i by N'_i such that $N_i \underset{d}{\rightarrow} N'_i$ at one step, or,

$M \underset{pd}{\rightarrow} N \iff M \equiv C[N_1, \ldots, N_n], N \equiv C[N'_1, \ldots, N'_n]$.

We say that the above $N_1, \ldots, N_n$ are $\underset{pd}{\rightarrow}$ redex occurrences. It is clear that $NF_{\underset{pd}{\rightarrow}} = NF_{\underset{d}{\rightarrow}}$. By the definition of parallel deletion reduction, one can easily prove that if $M \underset{pd}{\rightarrow} M'$ then $d(M) > d(M')$. Hence, every parallel deletion reduction terminates, that is, $SN(\underset{pd}{\rightarrow})$.

2.2.3.4. Lemma. Let $M \equiv C[\![A_1, \ldots, A_n]\!] \overset{M}{\rightarrow} C'[A_{i_1}, \ldots, A_{i_p}]$ where $1 \leq i_j \leq n$, and let $\langle A_1, \ldots, A_n\rangle \propto \langle B_1, \ldots, B_n\rangle$. Then one has a reduction $N \equiv C[B_1, \ldots, B_n] \overset{N}{\rightarrow} C'[B_{i_1}, \ldots, B_{i_p}]$.

Proof. The left side of the rewriting rule used in the reduction $\overset{M}{\rightarrow}$ occurs in

context $C[\ ,\ldots,\ ]$. Hence, one can apply this rewriting rule to N in the same way as for Lemma 2.2.2.4. □

2.2.3.5. Lemma. Let $d(M) > 1$, $M \equiv C[M_1, \ldots, M_m] \stackrel{M}{\to} C'[M_{i_1}, \ldots, M_{i_p}]$ $(1 \leq i_j \leq m\ )$, where $M_1, \ldots, M_m$ are all the $\underset{pd}{\to}$ redex occurrences in M. Let $\langle M_1, \ldots, M_m\rangle \propto \langle M'_1, \ldots, M'_m\rangle$. Then one has a reduction $M' \equiv C[M'_1, \ldots, M'_m] \stackrel{M'}{\to} C'[M'_{i_1}, \ldots, M'_{i_p}]$.

Proof. Let $M \equiv \tilde{C}[\![A_1, \ldots, A_n]\!]$, then $\forall i, \exists j, M_i \subseteq A_j$, and, thus, by replacing each M_i in A_j with M'_i, to make A'_j, one can obtain $M' \equiv \tilde{C}[A'_1, \ldots, A'_n]$. Now it is evident from $\langle M_1, \ldots, M_m\rangle \propto \langle M'_1, \ldots, M'_m\rangle$, that $\langle A_1, \ldots, A_n\rangle \propto \langle A'_1, \ldots, A'_n\rangle$. Hence Lemma 2.2.3.4 applies. □

2.2.3.6. Lemma. Let $d(M) > 1$, $M \equiv C[M_1, \ldots, M_m] \stackrel{M}{\to} C'[M_{i_1}, \ldots, M_{i_p}]$ $(1 \leq i_j \leq m)$, where $M_1, \ldots, M_m$ are all the $\underset{pd}{\to}$ redex occurrences in M. Let $\langle M_1, \ldots, M_m\rangle \stackrel{*}{\to} \langle M'_1, \ldots, M'_m\rangle$. Then one can obtain a term sequence $\langle M''_1, \ldots, M''_m\rangle$ such that $\langle M'_1, \ldots, M'_m\rangle \stackrel{*}{\to} \langle M''_1, \ldots, M''_m\rangle$ and $M' \equiv C[M''_1, \ldots, M''_m] \stackrel{M'}{\to} C'[M''_{i_1}, \ldots, M''_{i_p}]$.

Proof. In order to prove the lemma by using Lemma 2.2.3.5, we only need to find a $\langle M''_1, \ldots, M''_m\rangle$ such that $\langle M_1, \ldots, M_m\rangle \propto \langle M''_1, \ldots, M''_m\rangle$. Since $M_1, \ldots, M_m$ are $\underset{pd}{\to}$ redex occurrences, we obtain $\forall i, CR(M_i)$ by Theorem 2.2.2.14. Therefore, we obtain this $\langle M''_1, \ldots, M''_m\rangle$ by Lemma 2.2.2.10, taking $\alpha = \langle M_1, \ldots, M_m\rangle$, $\beta = \gamma = \langle M'_1, \ldots, M'_m\rangle$ and $\delta = \langle M''_1, \ldots, M''_m\rangle$. □

2.2.3.7. Lemma. Let $M \to N$, $M \underset{pd}{\to} P$, $d(M) = d(N)$. Then one has the diagram in Figure 2.13. Note that $d(M) > d(S)$.

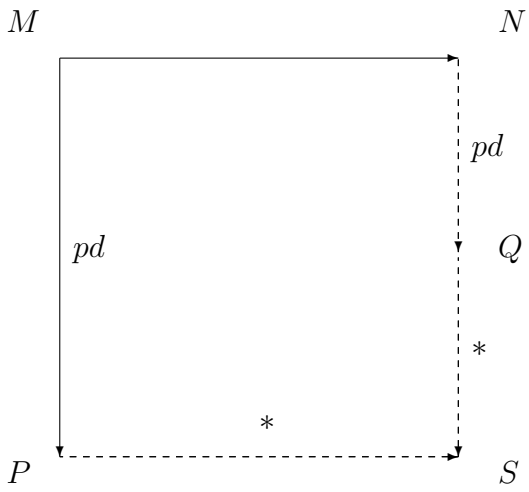


Figure 2.13

Proof. Let $M \xrightarrow{A} N$. The possible relative positions of the redex occurrence A and all of the $\underset{pd}{\rightarrow}$ redex occurrences in M, say $M_1, \ldots, M_m$, are given in the following cases.

Case 1. $\forall i, A \perp M_i$.

Then

$$\begin{aligned} M &\equiv C[M_1, \ldots, M_r, A, M_{r+1}, \ldots, M_m], \\ N &\equiv C[M_1, \ldots, M_r, B, M_{r+1}, \ldots, M_m], \\ P &\equiv C[P_1, \ldots, P_r, A, P_{r+1}, \ldots, P_m], \end{aligned}$$

where $A \xrightarrow{A} B$ and $\forall i, M_i \underset{d}{\rightarrow} P_i$. Since all of the $\underset{pd}{\rightarrow}$ redex occurrences in N are also $M_1, \ldots, M_m$ (this follows by $d(A) \geq d(B)$; A-contraction cannot create deeper $\underset{d}{\rightarrow}$ redex occurrences, in particular no $\underset{pd}{\rightarrow}$ redex occurrences), we can take $Q \equiv C[P_1, \ldots, P_r, B, P_{r+1}, \ldots, P_m]$. Let $S \equiv Q$, then $P \xrightarrow{*} S$ and $Q \xrightarrow{*} S$.

Case 2. $\exists r, A \subseteq M_r$.

Then

$$\begin{aligned} M &\equiv C[M_1, \ldots, M_{r-1}, M_r, M_{r+1}, \ldots, M_m], \\ N &\equiv C[M_1, \ldots, M_{r-1}, N_r, M_{r+1}, \ldots, M_m], \\ P &\equiv C[P_1, \ldots, P_{r-1}, P_r, P_{r+1}, \ldots, P_m], \end{aligned}$$

where $M_r \xrightarrow{A} N_r$, and $\forall i, M_i \underset{d}{\rightarrow} P_i$. Since each M_i $(i \neq r)$ is also a $\underset{pd}{\rightarrow}$ redex occurrence in N, by using $\underset{pd}{\rightarrow}$ for N, one obtains

$Q \equiv C[P_1, \ldots, P_{r-1}, Q_r, P_{r+1}, \ldots, P_m]$,

where $N_r \underset{d}{\overset{\equiv}{\rightarrow}} Q_r$, whether N_r is a $\underset{pd}{\rightarrow}$ redex occurrence or not (in N). By Theorem 2.2.2.14, $CR(M_r)$; therefore, there is a term S_r such that $P_r \xrightarrow{*} S_r$, $Q_r \xrightarrow{*} S_r$. Therefore, take

$S \equiv C[P_1, \ldots, P_{r-1}, S_r, P_{r+1}, \ldots, P_m]$.

Case 3. $\exists j, M_j \subset A$.

Let $M_r, \ldots, M_k$ $(r \leq k)$ be all the $\underset{pd}{\rightarrow}$ redex occurrences in M occurring in A. Then they are also $\underset{pd}{\rightarrow}$ redex occurrences in A. Let $A \equiv D[M_r, \ldots, M_k] \xrightarrow{A} D'[M_{i_1}, \ldots, M_{i_p}]$ $(r \leq i_j \leq k)$.

Then

$$\begin{aligned} M &\equiv C[M_1, \ldots, M_{r-1}, D[M_r, \ldots, M_k], M_{k+1}, \ldots, M_m], \\ N &\equiv C[M_1, \ldots, M_{r-1}, D'[M_{i_1}, \ldots, M_{i_p}], M_{k+1}, \ldots, M_m], \\ P &\equiv C[P_1, \ldots, P_{r-1}, D[P_r, \ldots, P_k], P_{k+1}, \ldots, P_m], \end{aligned}$$

where $\forall i, M_i \underset{d}{\rightarrow} P_i$. Since $M_1, \ldots, M_{r-1}, M_{k+1}, \ldots, M_m$ are also $\underset{pd}{\rightarrow}$ redex occurrences in N, whether $M_{i_1}, \ldots, M_{i_p}$ are $\underset{pd}{\rightarrow}$ redex occurrences or not (in N), one can obtain

$Q \equiv C[P_1, \ldots, P_{r-1}, D'[Q_{i_1}, \ldots, Q_{i_p}], P_{k+1}, \ldots, P_m]$,

where $\forall j, M_{i_j} \underset{d}{\overset{\equiv}{\rightarrow}} Q_{i_j}$. Now, by using Lemma 2.2.3.6, one can show for the subterm $D[P_r, \ldots, P_k]$ in P that there is a sequence $\langle P'_r, \ldots, P'_k \rangle$ such that $\langle P_r, \ldots, P_k \rangle \xrightarrow{*} \langle P'_r, \ldots, P'_k \rangle$ and $D[P'_r, \ldots, P'_k] \rightarrow D'[P'_{i_1}, \ldots, P'_{i_p}]$. Take

$P' \equiv C[P_1, \ldots, P_{r-1}, D'[P'_{i_1}, \ldots, P'_{i_p}], P_{k+1}, \ldots, P_m]$;

then one can have $P \xrightarrow{*} P'$. Since $\forall j, CR(M_{i_j})$, for each j there is S_{i_j} such that $P'_{i_j} \xrightarrow{*} S_{i_j}$, $Q_{i_j} \xrightarrow{*} S_{i_j}$. Therefore, take

$S \equiv C[P_1, \ldots, P_{r-1}, D'[S_{i_1}, \ldots, S_{i_p}], P_{k+1}, \ldots, P_m]$. □

2.2.3.8. Lemma. Let $M \to N$, $M \underset{pd}{\to} P$, $d(M) > d(N)$, then one has the diagram in Figure 2.14. Note that $d(M) > d(S)$.

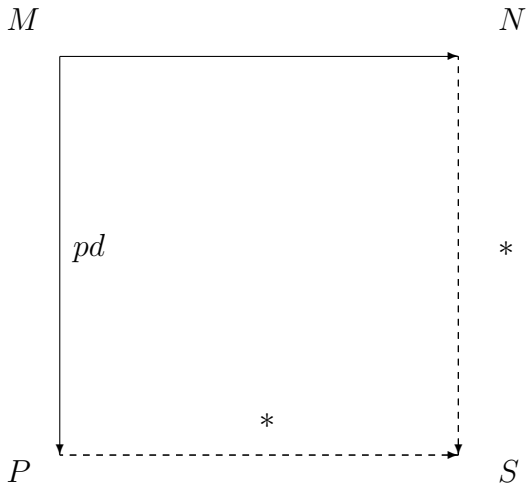


Figure 2.14

Proof. One can obtain a term S in the same way as for Case 2 and Case 3 in the proof of Lemma 2.2.3.7. □

2.2.3.9. Theorem. $R_1 \oplus R_2$ has the Church-Rosser property, that is, we have the diagram in Figure 2.15.

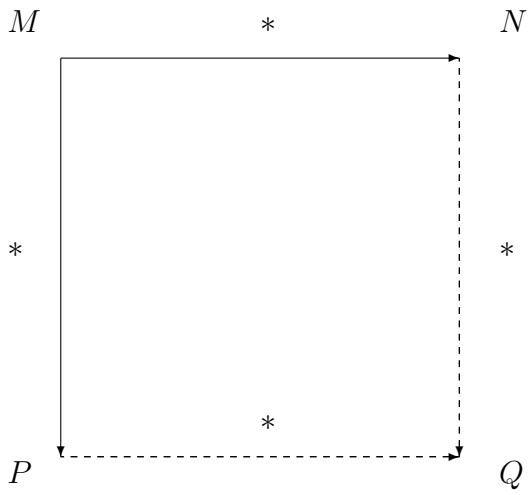


Figure 2.15

Proof. We will prove $CR(M)$ by induction on $d(M)$. The case $d(M) = 0$ is trivial from Theorem 2.2.2.13. Assume $CR(M)$ for $d(M) < n$ $(n > 0)$. Then we will show the following claim.

Claim. One has the diagram in Figure 2.16 for the case $d(M) \leq n$.

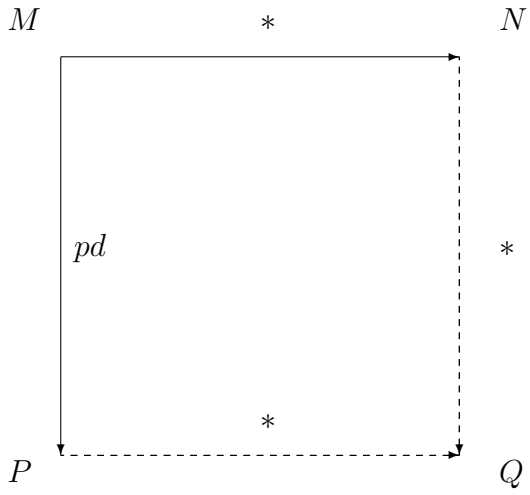


Figure 2.16

Proof of the Claim. Let $M \xrightarrow{m} N$, where $\xrightarrow{m}$ denotes a reduction of m $(m \geq 0)$

steps. Then we prove the claim by induction on m. The case $m = 0$ is trivial. Assume the claim for $m - 1$ ($m > 0$). We will show the diagram for m. Let $M \to A \overset{m-1}{\to} N$.

Case 1. $d(M) = d(A)$. We can obtain the diagram in Figure 2.17, proving diagram(1) by using Lemma 2.2.3.7, diagram(2) by using the induction hypothesis for the claim, and diagram(3) by using the induction hypothesis for the theorem, that is, $CR(B)$, since $d(M) > d(B)$.

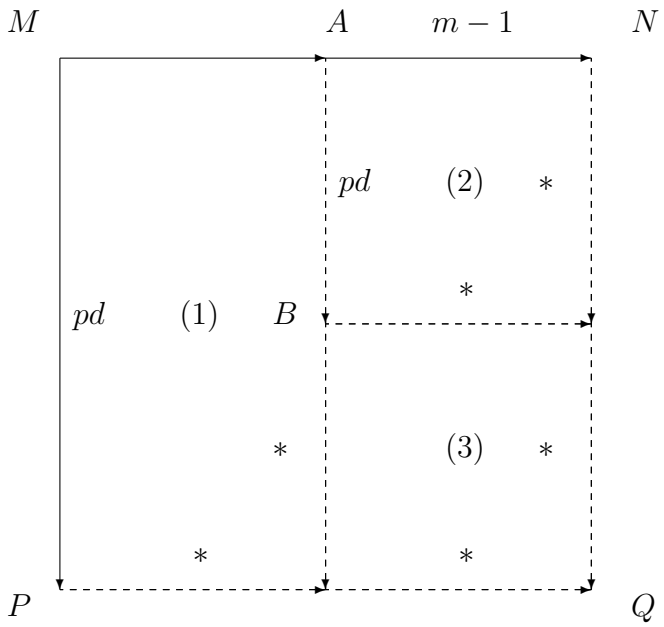


Figure 2.17

Case 2. $d(M) > d(A)$. We can obtain the diagram in Figure 2.18, proving diagram(1) by using Lemma 2.2.3.8, and diagram(2) by using the induction hypothesis for the theorem, that is, $CR(A)$.

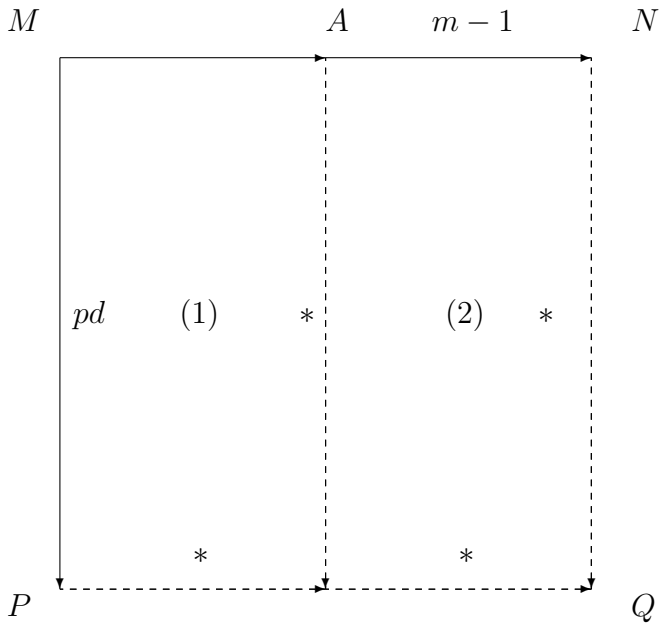


Figure 2.18

Now we will prove $CR(M)$ for $d(M) = n$. The diagram in Figure 2.19 can be obtained, where diagram(1) and diagram(2) are shown by the claim and the induction hypothesis, that is, $CR(A)$, respectively. □

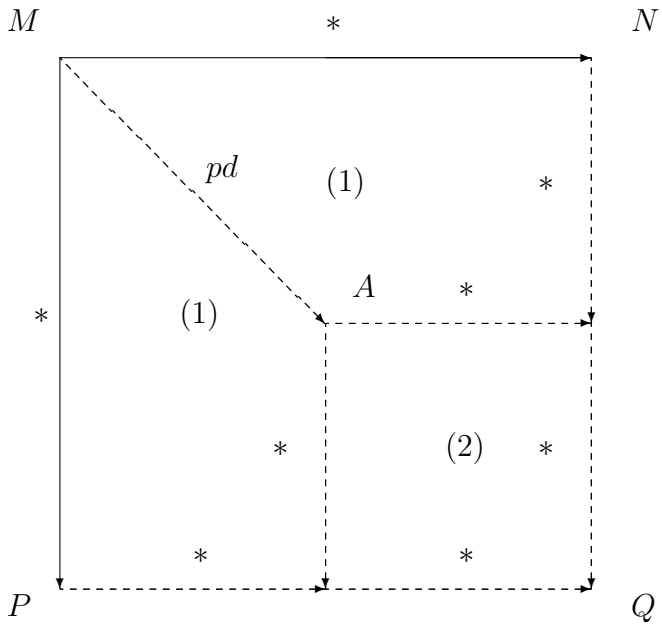


Figure 2.19

2.2.3.10. Corollary. $CR(R_1) \wedge CR(R_2) \iff CR(R_1 \oplus R_2)$.

Proof. $\Leftarrow$ is trivial, and $\Rightarrow$ is proved by Theorem 2.2.3.9. $\square$

2.3. Termination for the Direct Sum of Left-Linear Complete Term Rewriting Systems

In this section we consider the modular structure for the direct sum of left-linear term rewriting systems. The first result in this setting is due to Toyama [95] (see Section 2.2), where it is proven that confluence (i.e. the Church-Rosser property) is a modular property. To appreciate the non-triviality of this fact, it may be contrasted with the fact that another fundamental property, termination, is *not* modular, as the following simple counterexample in [96] shows:

$$R_0 \quad \left\{ \begin{array}{l} F(0,1,x) \triangleright F(x,x,x) \end{array} \right.$$

$$R_1 \quad \left\{ \begin{array}{l} g(x,y) \triangleright x \\ g(x,y) \triangleright y \end{array} \right.$$

It is trivial that R_0 and R_1 are terminating. However, $R_0 \oplus R_1$ is not terminating, because $R_0 \oplus R_1$ has the infinite reduction sequence:

$$F(g(0,1),g(0,1),g(0,1)) \to F(0,g(0,1),g(0,1)) \to f(0,1,g(0,1))$$

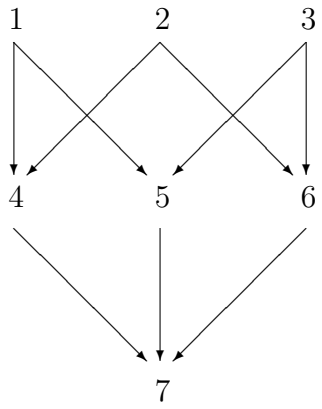
$$\to F(g(0,1),g(0,1),g(0,1)) \to \cdots.$$

The above counterexample uses a non-confluent term rewriting system R_1. A more complicated counterexample to the modularity of *termination*, involving only confluent term rewriting systems, was given by Klop and Barendregt [57, 96]. Consider R_0 and R_1 having the following rewriting rules:

$$R_0 \quad \left\{ \begin{array}{l} F(4,5,6,x) \triangleright F(x,x,x,x) \\ F(x,y,z,w) \triangleright 7 \\ 1 \triangleright 4 \\ 1 \triangleright 5 \\ 2 \triangleright 4 \\ 2 \triangleright 6 \\ 3 \triangleright 5 \\ 3 \triangleright 6 \\ 4 \triangleright 7 \\ 5 \triangleright 7 \\ 6 \triangleright 7 \end{array} \right.$$

$$R_1 \quad \begin{cases} g(x,x,y) \triangleright x \\ g(x,y,x) \triangleright x \\ g(y,x,x) \triangleright x \end{cases}$$

Note. R_0 has the following reductions:



Then, R_0 is confluent, because any term can be reduced into 7. R_0 is also terminating; no term can be reduced into 4, 5, and 6, hence, the first rule cannot be applied infinitely. Thus, R_0 is complete (a term rewriting system is complete iff it is both confluent and terminating). Clearly, R_1 is complete.

However, $R_0 \oplus R_1$ is not complete, since $F(M,M,M,M)$ with $M \equiv g(1,2,3)$ reduces to itself:

$$F(M,M,M,M) \to \cdots \to F(g(4,4,3), g(5,2,5), g(1,6,6), M) \to \cdots$$

$$\to F(4,5,6,M) \to F(M,M,M,M) \to \cdots.$$

This means that the important property of *completeness* of term rewriting systems is not modular. The counterexample, however, uses non-left-linear term rewriting systems.

The point of this section is that left-linearity is essential; if we restrict ourselves to left-linear term rewriting systems, then completeness is modular. Thus we prove: If R_0 and R_1 are left-linear (meaning that the rewriting rules have no repeated variables in their left-hand-sides), then R_0 and R_1 are complete iff $R_0 \oplus R_1$ is so. As left-linearity is a property which is so easily checked, and many equational algebraic specifications can be given by term rewriting systems which are left-linear, we feel that this result is worth while.

The proof, however, is rather intricate and not easily digested. A crucial element in the proof, and in general in the way that the summand term rewriting systems interact, is how terms may *collapse* to a subterm. The problem is that this collapsing behavior may exhibit a *nondeterministic* feature, which is caused by ambiguities among the rewriting rules. We propose the concept of the essential subterms for analyzing this nondeterministic collapsing behavior.

Regarding the question of modular properties in the present simple set-up, we mention the recent results by Rusinowitch [83] and Middeldorp [64]; these papers, together, contain a complete analysis of the cases in which termination for $R_0 \oplus R_1$ may be concluded from termination of R_0, R_1, depending on the distribution among R_0, R_1 of so-called collapsing and duplicating rules.

Another useful fact is established in Middeldorp [65], where it is proven that the *unique normal form property* is a modular property.

2.3.1. Preliminaries

The direct sum system $R_0 \oplus R_1$ is defined as the union of two term rewriting systems with disjoint function symbols (see Section 2.2). In this section, we assume that two disjoint systems R_0 on $T(F_0, V)$ and R_1 on $T(F_1, V)$ both are left-linear and complete (i.e., confluent and terminating). Then we shall prove that the direct sum system $R_0 \oplus R_1$ on $T(F_0 \cup F_1, V)$ is terminating. From here on the notation $\rightarrow$ represents

the reduction relation on $R_0 \oplus R_1$.

2.3.1.1. Lemma. $R_0 \oplus R_1$ is weakly normalizing, i.e., every term M has a normal form (denoted by $M\downarrow$).

Proof. Since R_0 and R_1 are terminating, M can be reduced into $M\downarrow$ through innermost reduction. □

We use again the following notations in Section 2.2 for this section.

Definition 2.2.1.1. A *root* of a term $M \in T(F_0 \cup F_1, V)$ is defined by

$$root(M) = \begin{cases} f & \text{if } M \equiv f(M_1, \ldots, M_n), \\ M & \text{if } M \text{ is a constant or a variable.} \end{cases}$$

Definition 2.2.1.2. Let $M \equiv C[B_1, \ldots, B_n] \in T(F_0 \cup F_1, V)$ and $C \not\equiv \Box$. Then write $M \equiv C[\![B_1, \ldots, B_n]\!]$ if $C[\ , \ldots, \ ]$ is a context on F_d and $\forall i, root(B_i) \in F_{\bar{d}}$ ($d \in \{0,1\}$ and $\bar{d} = 1 - d$). Then the set $S(M)$ of the special subterms of M is inductively defined as follows:

$$S(M) = \begin{cases} \{M\} & \text{if } M \in T(F_d, V) \ \ (d = 0 \text{ or } 1), \\ \bigcup_i S(B_i) \cup \{M\} & \text{if } M \equiv C[\![B_1, \ldots, B_n]\!] \ (n > 0). \end{cases}$$

The set of the special subterms having the root symbol in F_d is denoted by $S_d(M) = \{N |\ N \in S(M) \text{ and } root(N) \in F_d\}$.

Let $M \equiv C[\![B_1, \ldots, B_n]\!]$ and $M \xrightarrow{A} N$ (i.e., N results from M by contracting the redex occurrence A). If the redex occurrence A occurs in some B_j, then we write $M \underset{i}{\rightarrow} N$; otherwise $M \underset{o}{\rightarrow} N$. Here, $\underset{i}{\rightarrow}$ and $\underset{o}{\rightarrow}$ are called an inner and an outer reduction, respectively.

Definition 2.2.1.3. For a term $M \in T(F_0 \cup F_1, V)$, the rank of layers of contexts on F_0 and F_1 in M is inductively defined as follows:

$$rank(M) = \begin{cases} 1 & \text{if } M \in T(F_d, V) \ \ (d = 0 \text{ or } 1), \\ max_i\{rank(B_i)\} + 1 & \text{if } M \equiv C[\![B_1, \ldots, B_n]\!] \ (n > 0). \end{cases}$$

We often use Lemma 2.2.1.5 for the following discussions.

Lemma 2.2.1.5. If $M \to N$ then $rank(M) \geq rank(N)$.

2.3.1.2. Lemma. Let $M \xrightarrow{*} N$ and $root(M), root(N) \in F_d$. Then there exists a reduction $M \equiv M_0 \to M_1 \to M_2 \to \cdots \to M_n \equiv N \ \ (n \geq 0)$ such that $root(M_i) \in F_d$ for any i.

Proof. Let $M \xrightarrow{k} N \ \ (k \geq 0)$. We will prove the lemma by induction on k. The case $k = 0$ is trivial. Let $M \to M' \xrightarrow{k-1} N \ \ (k > 0)$. If $root(M') \in F_d$ then the lemma holds by the induction hypothesis. If $root(M') \in F_{\bar{d}}$ then there exists a context $C[\ ]$ with root $\in F_d$ such that $M \equiv C[M']$ and $C[\ ] \to \Box$. Thus, we can obtain a reduction $M \equiv C[M'] \xrightarrow{*} C[N] \to N$ in which all terms have root symbols in F_d. $\Box$

The set of terms in the reduction graph of M is denoted by $G(M) = \{N |\ M \xrightarrow{*} N\}$. The set of terms having the root symbol in F_d is denoted by $G_d(M) = \{N |\ N \in G(M) \ \text{ and } \ root(N) \in F_d\}$.

2.3.1.3. Definition. A term M is erasable iff $M \xrightarrow{*} x$ for some $x \in V$.

From now on we assume that every term $M \in T(F_0 \cup F_1, V)$ has only x as variable occurrences, unless it is stated otherwise. Since $R_0 \oplus R_1$ is left-linear, this variable convention may be assumed in the following discussions without loss of generality. If we need fresh variable symbols not in terms, we use z, z_1, z_2, $\cdots$.

2.3.2. Essential Subterms

In this subsection we introduce the concept of the essential subterms. We first prove the following property:

$$\forall N \in G_d(M)\ \exists P \in S_d(M),\ M \xrightarrow{*} P \xrightarrow{*} N.$$

2.3.2.1. Lemma. Let $M \to N$ and $Q \in S_d(N)$. Then, there exists some $P \in S_d(M)$ such that $P \xrightarrow{\equiv} Q$.

Proof. We will prove the lemma by induction on $rank(M)$. The case $rank(M) = 1$ is trivial. Assume the lemma for $rank(M) < k$ $(k > 1)$, then we will show the case $rank(M) = k$. Let $M \equiv C[\![M_1, \ldots, M_n]\!]$ $(n > 0)$ and $M \xrightarrow{A} N$.

Case 1. $M \equiv C[\![M_1, \ldots, M_r, \ldots, M_n]\!] \underset{o}{\xrightarrow{A}} N \equiv M_r$.
Then $S_d(N) \subseteq S_d(M)$.

Case 2. $M \equiv C[\![M_1, \ldots, M_n]\!] \underset{o}{\xrightarrow{A}} N \equiv C'[\![M_{i_1}, \ldots, M_{i_p}]\!]$ $(1 \le i_j \le n)$.
If $root(M) \in F_d$ then
$S_d(M) = \{M\} \cup \bigcup_i S_d(M_i)$,
$S_d(N) = \{N\} \cup \bigcup_j S_d(M_{i_j})$.
Thus the lemma holds since $\bigcup_j S_d(M_{i_j}) \subseteq \bigcup_i S_d(M_i)$, and $M \to N$.
If $root(M) \in F_{\bar{d}}$ then $S_d(N) = \bigcup_j S_d(M_{i_j}) \subseteq \bigcup_i S_d(M_i) = S_d(M)$.

Case 3. $M \equiv C[\![M_1, \ldots, M_r, \ldots, M_n]\!] \underset{i}{\xrightarrow{A}} N \equiv C[M_1, \ldots, M'_r, \ldots, M_n]$ where $M_r \xrightarrow{A} M'_r$.
If $root(M) \in F_d$ then
$S_d(M) = \{M\} \cup S_d(M_r) \cup \bigcup_{i \neq r} S_d(M_i)$,
$S_d(N) \subseteq \{N\} \cup S_d(M'_r) \cup \bigcup_{i \neq r} S_d(M_i)$.
If $root(M) \in F_{\bar{d}}$ then
$S_d(M) = S_d(M_r) \cup \bigcup_{i \neq r} S_d(M_i)$,

$S_d(N) = S_d(M'_r) \cup \bigcup_{i \neq r} S_d(M_i)$.

By the induction hypothesis, $\forall Q \in S_d(M'_r) \exists P \in S_d(M_r), P \stackrel{\equiv}{\rightarrow} Q$ for the both $root(M) \in F_d$ and $root(M) \in F_{\bar{d}}$. Thus the lemma holds. □

R_e consists of the single rule $e(x) \triangleright x$. $\underset{e}{\rightarrow}$ denotes the reduction relation of R_e, and $\underset{e'}{\rightarrow}$ denotes the reduction relation of $R_e \oplus (R_0 \oplus R_1)$ such that if $C[e(P)] \xrightarrow[e']{\Delta} N$ then the redex occurrence Δ does not occur in P. It is easy to show the confluence property of $\underset{e'}{\rightarrow}$.

From here on, $C[e(P_1), \cdots, e(P_p)]$ denotes a term such that $C[P_1, \cdots, P_p] \in T(F_0 \cup F_1, V)$, i.e., C and P_i contain no e.

2.3.2.2. Lemma. Let $C[e(P_1), \cdots, e(P_{i-1}), e(P_i), e(P_{i+1}), \cdots, e(P_p)] \xrightarrow[e']{k} e(P_i)$. Then $C[P_1, \cdots, P_{i-1}, e(P_i), P_{i+1}, \cdots, P_p] \xrightarrow[e']{k'} e(P_i)$ $(k' \leq k)$.

Proof. It is easily obtained from the definition and the left-linearity of the reduction $\underset{e'}{\rightarrow}$. □

Let $M \equiv C[P] \in T(F_0 \cup F_1, V)$ be a term containing no function symbol e. Now, consider $C[e(P)]$ by replacing the occurrence P in M with $e(P)$. Assume $C[e(P)] \xrightarrow[e']{*} e(P)$. Then, by tracing the reduction path, we can also obtain the reduction $M \equiv C[P] \xrightarrow{*} P$ (denoted by $M \xrightarrow[pull]{*} P$) under $R_0 \oplus R_1$. We say that the reduction $M \xrightarrow[pull]{*} P$ pulls up the occurrence P from M.

2.3.2.3. Example. Consider the two systems R_0 and R_1:

$$R_0 \quad \begin{cases} F(x) \triangleright G(x,x) \\ G(C,x) \triangleright x \end{cases}$$

$$R_1 \quad \begin{cases} h(x) \triangleright x \end{cases}$$

Then we have the reduction:

$F(e(h(C))) \underset{e'}{\rightarrow} G(e(h(C)), e(h(C))) \underset{e'}{\rightarrow} G(h(C), e(h(C))) \underset{e'}{\rightarrow} G(C, e(h(C))) \underset{e'}{\rightarrow} e(h(C))$.

Hence $F(h(C)) \xrightarrow[pull]{*} h(C)$. However, we cannot obtain $F(z) \xrightarrow[pull]{*} z$. Thus, in general, we cannot obtain $C[z] \xrightarrow[pull]{*} z$ from $C[P] \xrightarrow[pull]{*} P$. □

2.3.2.4. Lemma. Let $P \xrightarrow{*} Q$ and let $C[Q] \xrightarrow[pull]{*} Q$. Then $C[P] \xrightarrow[pull]{*} P$.

Proof. Let $M \equiv C[e(Q)] \xrightarrow[e']{k} e(Q)$. We will prove the lemma by induction on k. The case $k = 0$ is trivial. Let $M \equiv C[e(Q)] \underset{e'}{\longrightarrow} C'[e(Q), \cdots, e(Q), \cdots, e(Q)] \xrightarrow[e']{k-1} e(Q)$. Then, from Lemma 2.3.2.2 we can obtain the following reduction:

$C'[Q, \cdots, e(Q), \cdots, Q] \xrightarrow[e']{k'} e(Q)$ $(k' \leq k-1)$.

By using the induction hypothesis, $C'[Q, \cdots, e(P), \cdots, Q] \xrightarrow[e']{*} e(P)$. Therefore, we can obtain

$C[e(P)] \underset{e'}{\longrightarrow} C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{*} C'[Q, \cdots, e(P), \cdots, Q] \xrightarrow[e']{*} e(P)$

from $P \xrightarrow{*} Q$. □

2.3.2.5. Lemma. $\forall N \in G_d(M)\ \exists P \in S_d(M),\ M \xrightarrow[pull]{*} P \xrightarrow{*} N$.

Proof. If $root(M) \in F_d$ then the above property is trivial by taking M as P. Thus we consider only the non trivial case of $root(M) \in F_{\bar{d}}$. Let $M \xrightarrow{k} N$. We will prove the lemma by induction on k. The case $k = 1$ is trivial since $M \equiv C[\![M_1, \ldots, M_r, \ldots, M_n]\!] \rightarrow N \equiv M_r$ for some r (i.e., take $P \equiv M_r$). Assume the lemma for $k-1$. We will prove the case k. Let $M \rightarrow M' \xrightarrow{k-1} N$.

Case 1. $root(M') \in F_d$.

Then $M \equiv C[\![M_1, \ldots, M_r, \ldots, M_n]\!] \rightarrow M' \equiv M_r$ for some r. Take $P \equiv M_r$.

Case 2. $root(M') \in F_{\bar{d}}$.

By using the induction hypothesis, $\exists P' \in S_d(M'),\ M' \xrightarrow[pull]{*} P' \xrightarrow{*} N$. Here, from Lemma 2.3.2.1, there exists some $P \in S_d(M)$ such that $P \xrightarrow{\equiv} P'$. We will consider the following two subcases:

Case 2.1. $P \rightarrow P'$. Then $M \equiv C[P] \rightarrow M' \equiv C[P']$. Thus, by using Lemma 2.3.2.4, $M \equiv C[P] \xrightarrow[pull]{*} P \rightarrow P' \xrightarrow{*} N$.

Case 2.2. $P \equiv P'$. Then, for some context $C'[\ , \cdots,\ ]$, $M \equiv C[P] \rightarrow M' \equiv C'[P, \cdots, P, \cdots, P]$ and $C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{*} e(P)$. Therefore

$C[e(P)] \rightarrow C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e]{*} C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{*} e(P)$. Thus $M \equiv C[P] \xrightarrow[pull]{*} P \xrightarrow{*} N$. □

Now, we introduce the concept of the essential subterms. The set $E_d(M)$ of the essential subterms of the term $M \in T(F_0 \cup F_1, V)$ is defined as follows:

$E_d(M) = \{P \mid M \xrightarrow[pull]{*} P \in S_d(M)$ and $\neg\exists Q \in S_d(M)\ [M \xrightarrow[pull]{*} Q \xrightarrow{+} P]\}$.

The following lemmas are easily obtained from the definition of the essential subterms and Lemma 2.3.2.5.

2.3.2.6. Lemma. $\forall N \in G_d(M)\ \exists P \in E_d(M),\ P \xrightarrow{*} N$.

2.3.2.7. Lemma. $E_d(M) = \phi$ iff $G_d(M) = \phi$.

We say M is deterministic for d if $|E_d(M)| = 1$; M is nondeterministic for d if $|E_d(M)| \geq 2$. The following lemma plays an important role in the next subsection.

2.3.2.8. Lemma. If $root(M\downarrow) \in F_d$ then $|E_d(M)| = 1$, i.e., M is deterministic for d.

Proof. See Appendix A. □

2.3.3. Termination for the Direct Sum

In this subsection we will show that $R_0 \oplus R_1$ is terminating. Roughly speaking, termination is proved by showing that any infinite reduction $M_0 \to M_1 \to M_2 \to \cdots$ of $R_0 \oplus R_1$ can be translated into an infinite reduction $M'_0 \to M'_1 \to M'_2 \to \cdots$ of R_d.

We first define the term $M^d \in T(F_d, V)$ for any term M and any d.

2.3.3.1. Definition. For any M and any d, $M^d \in T(F_d, V)$ is defined by induction on $rank(M)$:

(1) $M^d \equiv M$ if $M \in T(F_d, V)$.

(2) $M^d \equiv x$ if $E_d(M) = \phi$.

(3) $M^d \equiv C[M_1^d, \cdots, M_m^d]$ if $root(M) \in F_d$ and $M \equiv C[\![M_1, \cdots, M_m]\!]$ $(m > 0)$.

(4) $M^d \equiv P^d$ if $root(M) \in F_{\bar{d}}$ and $E_d(M) = \{P\}$. Note that $rank(P) < rank(M)$.

(5) $M^d \equiv C_1[C_2[\cdots C_{p-1}[C_p[x]]\cdots]]$ if $root(M) \in F_{\bar{d}}$, $E_d(M) = \{P_1, \cdots, P_p\}$ $(p > 1)$, and every P_i^d is erasable. Here $P_i^d \equiv C_i[x] \xrightarrow[pull]{*} x$ $(i = 1, \cdots, p)$. Note that, for any i, $rank(P_i) < rank(M)$ and $M^d \xrightarrow{*} P_i^d$.

(6) $M^d \equiv x$ if $root(M) \in F_{\bar{d}}$, $|E_d(M)| \geq 2$, and not (5).

Note that M^d is not unique if a subterm of M^d is constructed with (5) in the above definition.

2.3.3.2. Lemma. $root(M\downarrow) \notin F_d$ iff $M^d \downarrow \equiv x$.

Proof. Instead of the lemma, we will prove the following claim:

Claim. If $root(M\downarrow) \notin F_d$ then $M^d \downarrow \equiv x$. If $root(M\downarrow) \in F_d$ and $M\downarrow \equiv \hat{C}[\![M_1, \cdots, M_m]\!]$ then $M^d \downarrow \equiv \hat{C}[x, \cdots, x]$.

Proof of the Claim. We will prove the lemma by induction on $rank(M)$. The case $rank(M) = 1$ is trivial by the definition of M^d. Assume the lemma for $rank(M) < k$ $(k \geq 2)$. Then we will prove the case $rank(M) = k$.

Case 1. $root(M) \in F_d$.

Let $M \equiv C[\![M_1, \cdots, M_m]\!]$. Then $M^d \equiv C[M_1^d, \cdots, M_m^d]$. We may assume that $root(M_i\downarrow) \notin F_d$ $(1 \leq i < p)$ and $root(M_j\downarrow) \in F_d$ $(p \leq j \leq m)$ without loss of generality. Let $M_j \downarrow \equiv \hat{C}_j[\![N_{j,1}, \cdots, N_{j,n_j}]\!]$ $(p \leq j \leq m)$. Then, by using the induction hypothesis, $M_i^d \downarrow \equiv x$ $(1 \leq i < p)$ and $M_j^d \downarrow \equiv \hat{C}_j[x, \cdots, x]$ $(p \leq j \leq m)$.
Thus $M\downarrow \equiv C[M_1\downarrow, \cdots, M_m\downarrow]\downarrow$
$\equiv C[M_1\downarrow, \cdots, M_{p-1}\downarrow, \hat{C}_p[\![N_{p,1}, \cdots, N_{p,n_p}]\!], \cdots, \hat{C_m}[\![N_{m,1}, \cdots, N_{m,n_m}]\!]]\downarrow$
and $M^d \downarrow \equiv C[M_1^d\downarrow, \cdots, M_m^d\downarrow]\downarrow \equiv C[x, \cdots, x, \hat{C}_p[x, \cdots, x], \cdots, \hat{C_m}[x, \cdots, x]]\downarrow$.
Note that $M_i\downarrow$ $(1 \leq i < p)$, $N_{p,1}, \cdots, N_{m,n_m}$ are normal forms having root symbols not in F_d. Therefore, if $root(M\downarrow) \notin F_d$ then $C[x, \cdots, x, \hat{C}_p[x, \cdots, x], \cdots, \hat{C_m}[x, \cdots, x]]\downarrow \equiv x$; if $root(M\downarrow) \in F_d$ then we have a context
$\hat{C}[\,, \cdots, \,] \equiv C[\,, \cdots, \,, \hat{C}_p[\,, \cdots, \,], \cdots, \hat{C_m}[\,, \cdots, \,]]\downarrow$ such that $M\downarrow \equiv \hat{C}[\![N_1, \cdots, N_n]\!]$ where $N_i \in \{M_1\downarrow, \cdots, M_{p-1}\downarrow, N_{p,1}, \cdots, N_{m,n_m}\}$ and $M^d \downarrow \equiv \hat{C}[x, \cdots, x] \not\equiv x$.

Case 2. $root(M) \notin F_d$.

Consider three subcases:

Case 2.1. $E_d(M) = \phi$.

From Lemma 2.3.2.7, $root(M\downarrow) \notin F_d$. Since $M^d \equiv x$, $M^d \downarrow\equiv x$.

Case 2.2. $E_d(M) = \{P\}$.

Then $M^d \equiv P^d$. Note that $rank(P) < k$. Since $M\downarrow\equiv P\downarrow$ and $M^d\downarrow\equiv P^d\downarrow$, the claim follows by using the induction hypothesis.

Case 2.3. $E_d(M) = \{P_1, \cdots, P_p\}$ $(p > 1)$.

Note that $rank(P_i) < k$ for any i. From Lemma 2.3.2.8, $root(M\downarrow) \notin F_d$. Since $M\downarrow\equiv P_i\downarrow$, it is clear that $root(P_i\downarrow) \notin F_d$ for all i. Thus, we have $P_i^d\downarrow\equiv x$ by the induction hypothesis. From case (5) in the definition of M^d, it follows that $M^d\downarrow\equiv x$. □

Note. Let $E_d(M) = \{P_1, \cdots, P_p\}$ $(p > 1)$. Then, from Lemma 2.3.2.8 and Lemma 2.3.3.2, it follows that every P_i^d is erasable. Hence case (6) in the definition of M^d can be removed.

2.3.3.3. Lemma. If $P \in E_d(M)$ then $M^d \xrightarrow{*} P^d$.

Proof. Obvious from the definition of M^d and the above note. □

We wish to translate directly an infinite reduction $M_0 \to M_1 \to M_2 \to \cdots$ into an infinite reduction $M_0^d \xrightarrow{*} M_1^d \xrightarrow{*} M_2^d \xrightarrow{*} \cdots$. However, the following example shows that $M_i \to M_{i+1}$ cannot be translated into $M_i^d \xrightarrow{*} M_{i+1}^d$ in general.

2.3.3.4. Example. Consider the two systems R_0 and R_1:

$$R_0 \quad \begin{cases} F(C, x) \triangleright x \\ F(x, C) \triangleright x \end{cases}$$

$$R_1 \quad \begin{cases} f(x) \triangleright g(x) \\ f(x) \triangleright h(x) \\ g(x) \triangleright x \\ h(x) \triangleright x \end{cases}$$

Let $M \equiv F(f(C), h(C)) \to N \equiv F(g(C), h(C))$. Then $E_1(M) = \{f(C)\}$ and $E_1(N) = \{g(C), h(C)\}$. Thus $M^1 \equiv f(x)$, $N^1 \equiv g(h(x))$. It is obvious that $M^1 \xrightarrow{*} N^1$ does not hold. □

Now we will consider to translate indirectly an infinite reduction of $R_0 \oplus R_1$ into an infinite reduction of R_d.

We write $M \underset{o}{\equiv} N$ when M and N have the same outermost-layer context, i.e., $M \equiv C[\![M_1, \cdots, M_m]\!]$ and $N \equiv C[\![N_1, \cdots, N_m]\!]$ for some $M_i,\ N_i$.

2.3.3.5. Lemma. Let $A \underset{i}{\xrightarrow{*}} M$, $M \underset{o}{\to} N$, $A \underset{o}{\equiv} M$, and $root(M), root(N) \in F_d$. Then, for any A^d there exist B and B^d such that

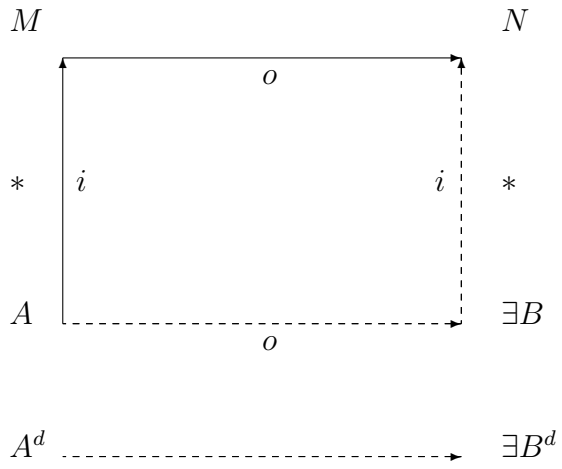


Figure 2.20

Proof. Let $A \equiv C[\![A_1, \cdots, A_m]\!]$, $M \equiv C[\![M_1, \cdots, M_m]\!]$, $N \equiv C'[\![M_{i_1}, \cdots, M_{i_n}]\!]$ $(i_j \in \{1, \cdots, m\})$. Take $B \equiv C'[\![A_{i_1}, \cdots, A_{i_n}]\!]$. Then, we can obtain $A \underset{o}{\rightarrow} B$ and $B \underset{i}{\overset{*}{\rightarrow}} N$. From $A^d \equiv C[A_1^d, \cdots, A_m^d]$ and $B^d \equiv C'[A_{i_1}^d, \cdots, A_{i_n}^d]$, it follows that $A^d \rightarrow B^d$. $\Box$

2.3.3.6. Lemma. Let $M \overset{*}{\rightarrow} N$, $root(N) \in F_d$. Then, for any M^d there exist A $(A \underset{o}{\equiv} N)$ and A^d such that

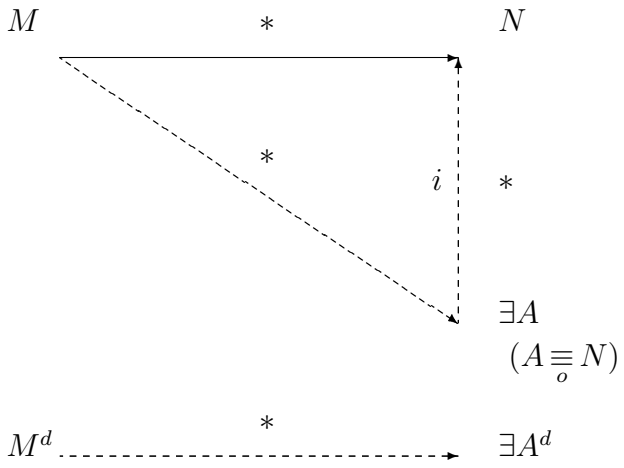


Figure 2.21

Proof. We will prove the lemma by induction on $rank(M)$. The case $rank(M) = 1$ is trivial by taking $A \equiv N$. Assume the lemma for $rank(M) < k$. Then we will prove the case $rank(M) = k$. We start from the following claim.

Claim. The lemma holds if $M \underset{i}{\overset{*}{\rightarrow}} N$.

Proof of the Claim. Let $M \equiv C[\![M_1, \cdots, M_m]\!] \underset{i}{\overset{*}{\rightarrow}} N \equiv C[N_1, \cdots, N_m]$ where $M_i \overset{*}{\rightarrow} N_i$ for every i. We may assume that $N_1 \equiv x, \cdots, N_{p-1} \equiv x$, $root(N_i) \in F_d$ $(p \leq i \leq q-1)$, and $root(N_j) \in F_{\bar{d}}$ $(q \leq j \leq m)$ without loss of generality.

Thus $N \equiv C[x, \cdots, x, N_p, \cdots, N_{q-1}, N_q, \cdots, N_m]$. Then, by using the induction hypothesis, every M_i $(p \leq i \leq q-1)$ has A_i $(A_i \underset{o}{\equiv} N_i)$ and A_i^d such that

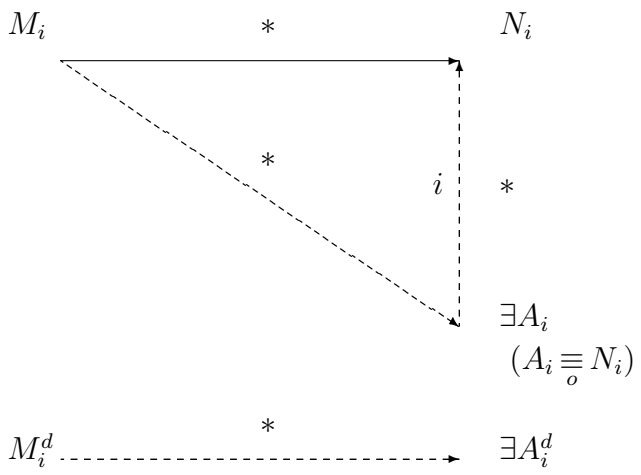


Figure 2.22

Now, take $A \equiv C[x, \cdots, x, A_p, \cdots, A_{q-1}, M_q, \cdots, M_m]$. It is obvious that $M \xrightarrow{*} A$. From Lemma 2.3.1.2, we can have the reductions $M_j \xrightarrow{*} N_j$ $(q \leq j \leq m)$ in which every term has a root symbol in $F_{\bar{d}}$. Thus it follows that $A \underset{i}{\xrightarrow{*}} N$ and $A \underset{o}{\equiv} N$. From Lemma 2.3.3.2 and $M_i \downarrow\equiv x$ $(1 \leq i < p)$, $M_i^d \downarrow\equiv x$. Therefore, since

$M^d \equiv C[M_1^d, \cdots, M_{p-1}^d, M_p^d, \cdots, M_{q-1}^d, M_q^d, \cdots, M_m^d]$

and $A^d \equiv C[x, \cdots, x, A_p^d, \cdots, A_{q-1}^d, M_q^d, \cdots, M_m^d]$, it follows that $M^d \xrightarrow{*} A^d$. *(end of the claim)*

Now we will prove the lemma for $rank(M) = k$. Consider two cases.

Case 1. $root(M) \in F_d$.

From Lemma 2.3.1.2, we may assume that every term in the reduction $M \xrightarrow{*} N$ has a root symbol in F_d. By splitting $M \xrightarrow{*} N$ into $M \underset{i}{\xrightarrow{*}} \underset{o}{\rightarrow} \underset{i}{\xrightarrow{*}} \underset{o}{\rightarrow} \cdots \underset{i}{\xrightarrow{*}} N$ and using the claim for diagram (1) and Lemma 2.3.3.5 for diagram (2), we can draw the

following diagram:

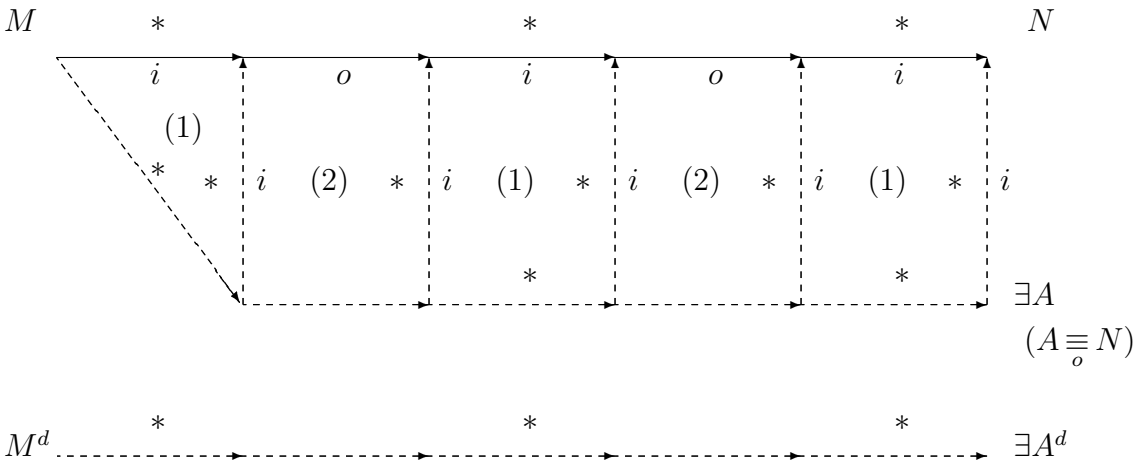


Figure 2.23

Note that if $M' \xrightarrow[i]{*} M'' \xrightarrow[i]{*} M'''$ then $M' \xrightarrow[i]{*} M'''$; thus, the claim can be applied to diagram (1) in the above diagram.

Case 2. $root(M) \in F_{\bar{d}}$.

Then we have some essential subterm $Q \in E_d(M)$ such that $M \xrightarrow{*} Q \xrightarrow{*} N$. From Lemma 2.3.3.3, it follows that $M^d \xrightarrow{*} Q^d$. It is obvious that $rank(Q) < k$. Hence, we have the following diagram, where diagram (1) is obtained by the induction hypothesis:

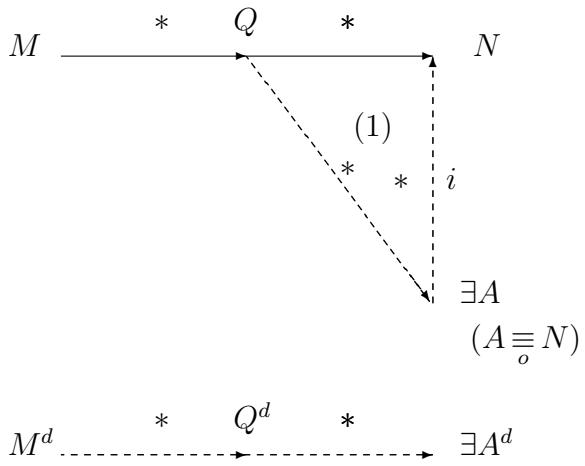


Figure 2.24

□

Now we can prove the following theorem:

2.3.3.7. Theorem. No term M has an infinite reduction.

Proof. We will prove the theorem by induction on $rank(M)$. The case $rank(M) = 1$ is trivial. Assume the theorem for $rank(M) < k$. Then, we will show the case $rank(M) = k$. Suppose M has an infinite reduction $M \rightarrow\rightarrow\rightarrow \cdots$. From the induction hypothesis, we can have no infinite inner reduction $\underset{i}{\rightarrow}\underset{i}{\rightarrow}\underset{i}{\rightarrow}\cdots$ in this reduction. Thus, $\underset{o}{\rightarrow}$ must infinitely appear in the infinite reduction. From the induction hypothesis, all of the terms appearing in this reduction have the same rank; hence, their root symbols are in F_d if $root(M) \in F_d$. Hence, by a similar construction of diagrams as for Case 1 in the proof of Lemma 2.3.3.6, it follows that M^d has an infinite reduction. This contradicts that R_d is terminating. □

2.3.3.8. Corollary Two term rewriting systems R_0 and R_1 are left-linear and

complete iff the direct sum $R_0 \oplus R_1$ is so.

Proof. $\Leftarrow$ is trivial. $\Rightarrow$ follows from Theorem 2.3.3.7 and Corollary 2.2.3.10 in Section 2.2 stating that two term rewriting systems R_0 and R_1 are confluent iff the direct sum $R_0 \oplus R_1$ is so. $\Box$

2.4. Conclusion

In this chapter, we have proven that the Church-Rosser property is modular [95]: term rewriting systems R_1 and R_2 are Church-Rosser iff the direct sum $R_1 \oplus R_2$ is so. The strength of the property lies in the absence of assumptions about R_1 and R_2. To appreciate the non-triviality of this fact, it has been contrasted with the fact that the termination property is not modular [96]. Finally, we have shown that the completeness (i.e., Church-Rosser and termination) property is also modular under left-linearity: two term rewriting systems R_1 and R_2 are left-linear and complete iff $R_1 \oplus R_2$ is so [100, 99]. We think that the modularities presented here will be essential building blocks of the theory of term rewriting systems.

To conclude this chapter, we mention the recent works. Starting with our research [95] (Section 2.2), various modularities have been studied by several authors. Sufficient conditions for the modular property of termination were studied by Rushinowitch [83], Middeldorp [64], Toyama, Klop and Barendregt [100, 99] (Section 2.3). Middeldorp showed that the unique normal form property is modular, but the normal form property is not [65]. Furthermore, these modularities were extended to conditional term rewriting systems by Middeldorp [66, 67, 68]. Kurihara and Kaiji proposed an alternative approach to modularity [61]. The modularities for the combinations of the lambda calculus and term rewriting systems were studied by Breazu-Tannen [6], Breazu-Tannen and Gallier [7], and Dougherty [23].

3. Commutativity of Term Rewriting Systems

Commutativity is very useful in showing the Church-Rosser property for the union of term rewriting systems. In this chapter we study the critical pair technique for proving commutativity of term rewriting systems. Extending the concept of critical pairs between two term rewriting systems, a sufficient condition for commutativity is proposed. Using the proposed result, a new sufficient condition is offered for the Church-Rosser property of left-linear term rewriting systems.

3.1. Introduction

We consider the commutative property of two term rewriting systems R_1 and R_2 [81]. Hindley [34] and Rosen [81] first studied commutative reduction systems by considering how to infer the Church-Rosser property for a complex system from various properties of its parts. They showed that if R_1 and R_2 commute and have the Church-Rosser property, then the union $R_1 \cup R_2$ also has the Church-Rosser property.

Simple sufficient conditions for commutativity or quasi-commutativity of linear term rewriting systems R_1 and R_2 have been proposed [1, 44, 46, 79, 88]: For example, if two left-linear term rewriting systems R_1 and R_2 do not overlap, then they commute [79, 88]. However, these works were done on the following restrictions: R_1 and R_2 are nonoverlapping with each other [1, 79, 88], or R_1 is $(E-)$ terminating

[44, 46]. Hence new conditions are needed to prove commutativity if the systems do not satisfy these restrictions.

In this chapter we study commutativity of left-linear term rewriting systems R_1 and R_2 without the above restrictions. That is, two systems may overlap and be nonterminating. To treat the overlapping and terminating case, the critical pair concept used to infer the Church-Rosser property [37, 59, 81] is extended. This extension is done by introducing the critical pairs between R_1 and R_2 and classifying them into two kinds of pairs; outside pairs and inside pairs. These extended critical pairs are used to propose a sufficient condition for commutativity of term rewriting systems. The proposed result can also be applied to inferring the Church-Rosser property. A new sufficient condition is offered for the Church-Rosser property of left-linear term rewriting systems with overlapping rules.

3.2. Extended Critical Pairs

The critical pair concept (see Section 1.4) for a term rewriting system will be extended into a concept for two systems. Let R_1 and R_2 be two term rewriting systems and let $P \triangleright Q \in R_1$ and $M \triangleright N \in R_2$. It may be assumed that the variables have been renamed appropriately, so that P and M share no variables. Assume $S \notin V$ is a subterm occurrence in M, i.e. $M \equiv C[S]$, such that S and P are unifiable, i.e. $S\theta \equiv P\theta$, with a minimal unifier θ [37, 59]. Since $M\theta \equiv C[S]\theta \equiv C\theta[P\theta]$, two reductions starting with $M\theta$, i.e. $M\theta \underset{1}{\rightarrow} C\theta[Q\theta] \equiv C[Q]\theta$ and $M\theta \underset{2}{\rightarrow} N\theta$, can be obtained using $P \triangleright Q \in R_1$ and $M \triangleright N \in R_2$ respectively. Then $P \triangleright Q$ is said to overlap $M \triangleright N$, and the pair of terms $\langle C[Q]\theta, N\theta\rangle$ is a critical pair of $P \triangleright Q$ on $M \triangleright N$. The pair is inside (resp. outside) critical if $S \subset M$ (resp. $S \equiv M$). $P \triangleright Q \in R_1$ and $M \triangleright N \in R_2$ may be chosen to be the same rule, but in this case we shall not consider the case $S \equiv M$, which gives the trivial pair $\langle N, N\rangle$. Note that two rules play asymmetrical role in this definition.

$crit(R_1, R_2)$ denotes the set of the critical pairs for all $P \triangleright Q \in R_1$ and $M \triangleright N \in R_2$ such that $P \triangleright Q$ overlaps $M \triangleright N$. $crit_{in}(R_1, R_2)$ and $crit_{out}(R_1, R_2)$ denote the set of inside critical pairs and the set of outside critical pairs respectively. Thus $crit(R_1, R_2) = crit_{in}(R_1, R_2) \cup crit_{out}(R_1, R_2)$. Note that generally $crit(R_1, R_2) \neq crit(R_2, R_1)$ since the definition of overlapping is asymmetrical.

$crit(R)$, $crit_{in}(R)$ and $crit_{out}(R)$ indicate $crit(R, R)$, $crit_{in}(R, R)$ and $crit_{out}(R, R)$ respectively. Thus $crit(R)$ coincides with the set of critical pairs of R defined in [37, 59].

We say that R_1 and R_2 are overlapping with each other if $crit(R_1, R_2) \cup crit(R_2, R_1) \neq \phi$; R_1 and R_2 are nonoverlapping with each other if they are not overlapping with each other. R is overlapping if $crit(R) \neq \phi$; R is nonoverlapping if it is not overlapping. [37, 59].

Remark. Jouannaud and Kirchner [44] and Jouannaud and Munoz [46] also proposed the idea of critical pairs between two systems R_1 and R_2 independently of the author. However, they applied it in a different situation, to discuss the sufficient conditions for the Church-Rosser property and for the termination property of $R_1 \cup R_2$ under the stronger assumptions that R_1 is E-terminating and R_2 is an equational system E. This paper does not assume the termination property of term rewriting systems.

The following sufficient conditions for the Church-Rosser property are stated in Chapter 1.4.

Proposition 1.4.8. (Knuth-Bendix). Let R be strongly normalizing. Then R has the Church-Rosser property iff P and Q have the same normal form for any critical pair $\langle P, Q \rangle$ in R.

Proposition 1.4.9. (Rosen). Let R be left-linear and non-overlapping. Then R has the Church-Rosser property.

Proposition 1.4.11 (Huet). Let R be left-linear. If $P \twoheadrightarrow Q$ for every critical pair $\langle P, Q\rangle$ in R, then R has the Church-Rosser property.

For more discussion concerning the Church-Rosser property of term rewriting systems, see [37, 44, 72, 95].

3.3. Sufficient Condition for Commutativity

This section shows a sufficient condition for commutativity of two left-linear term rewriting systems R_1 and R_2 on $T(F, V)$. From here on, $\underset{i}{\rightarrow}$ and $\underset{i}{\twoheadrightarrow}$ denote the reduction relation and the parallel reduction relation of R_i $(i = 1, 2)$ respectively.

3.3.1. Lemma. If we have the diagram in Figure 3.1 then R_1 commutes with R_2.

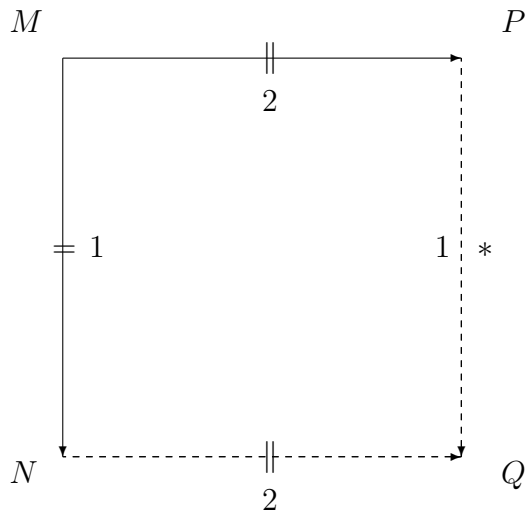


Figure 3.1

Proof. From $\overset{*}{\underset{1}{\dashrightarrow}} = \overset{*}{\underset{1}{\rightarrow}}$, we obtain

$\forall M, N, P[M \overset{}{\underset{1}{\dashrightarrow}} N \wedge M \overset{}{\underset{2}{\dashrightarrow}} P \Rightarrow \exists Q, N \overset{}{\underset{2}{\dashrightarrow}} Q \wedge P \overset{*}{\underset{1}{\dashrightarrow}} Q]$.

By applying Proposition 1.3.7 (Commutativity Lemma), we can prove commutativity of $\overset{*}{\underset{1}{\dashrightarrow}}$ and $\overset{*}{\underset{2}{\dashrightarrow}}$. Since $\overset{*}{\underset{i}{\dashrightarrow}} = \overset{*}{\underset{i}{\rightarrow}}$ $(i = 1, 2)$, it follows that R_1 commutes with R_2. □

Let $A \equiv C[x_1, \cdots, x_n]$ where no variable occurs in C. Then we say the subterm occurrence P of $A\theta \equiv C[x_1\theta, \cdots, x_n\theta]$ occurs in the substitution θ if P occurs in some $x_i\theta$.

3.3.2. Lemma. Let $M \equiv A\theta \underset{1}{\overset{M}{\longrightarrow}} N \equiv B\theta$, $A \triangleright B \in R_1$, and $M \equiv A\theta \overset{P_1,\cdots,P_p}{\underset{2}{\dashrightarrow}} P$ where P_i $(i = 1, \cdots, p)$ occurs in θ. Then a term Q can be obtained such that $N \overset{}{\underset{2}{\dashrightarrow}} Q$ and $P \underset{1}{\longrightarrow} Q$ (Figure 3.2).

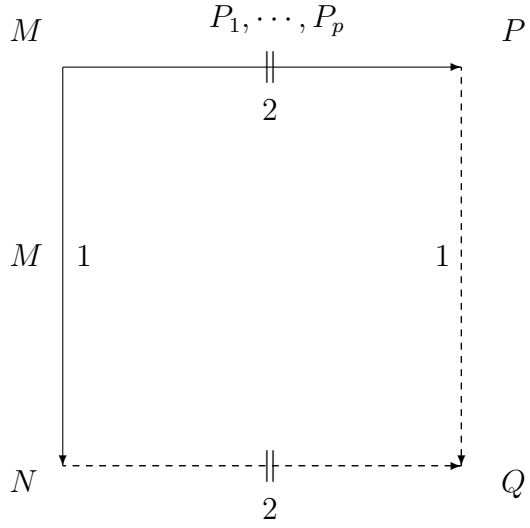


Figure 3.2

Proof. Since P_i $(i = 1, \cdots, p)$ occurs in θ, $P \equiv A\theta'$ can be denoted for some θ' such that $x\theta \underset{2}{\twoheadrightarrow} x\theta'$ for any x in A. Take $Q \equiv B\theta'$. Then it follows that $N \equiv B\theta \underset{2}{\twoheadrightarrow} Q \equiv B\theta'$ and $P \equiv A\theta' \underset{1}{\longrightarrow} Q \equiv B\theta'$. □

3.3.3. Theorem. Let R_1 and R_2 be left-linear term rewriting systems. Then R_1 commutes with R_2 if R_1 and R_2 satisfy the following conditions:

(1) $\forall \langle P, Q\rangle \in crit(R_1, R_2)\ \exists S\ [P \underset{2}{\twoheadrightarrow} S \wedge Q \underset{1}{\overset{*}{\rightarrow}} S]$,

(2) $\forall \langle Q, P\rangle \in crit_{in}(R_2, R_1)\ \ [Q \underset{1}{\twoheadrightarrow} P]$.

Proof. Let $M \overset{A_1,\cdots,A_m}{\underset{1}{\twoheadrightarrow}} N$ and $M \overset{B_1,\cdots,B_n}{\underset{2}{\twoheadrightarrow}} P$. If we have the diagram in Figure 3.3, then the theorem follows from Lemma 3.3.1. Hence we will show the existence of the term Q in Figure 3.3 under the above conditions.

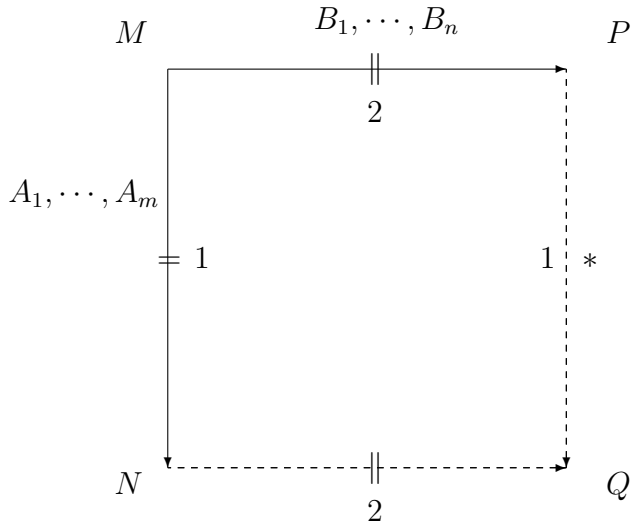


Figure 3.3

Let $\Gamma = \{A_i | \exists B_j, A_i \subseteq B_j\} \cup \{B_i | \exists A_j, B_i \subseteq A_j\}$ and $\Delta = \{A_i | \forall B_j, A_i \not\subseteq B_j\} \cup \{B_i | \forall A_j, B_i \not\subseteq A_j\}$. Then the redex occurrences $A_1, \cdots, A_m$ and $B_1, \cdots, B_n$ of M are classified into two sets Γ and Δ. The length $|M|$ of a term M is defined by the number of symbols in M. $|\Gamma|$ denotes $\sum_{M \in \Gamma} |M|$. By using induction on $|\Gamma|$, we will prove the existence of Q in Figure 3.3.

The case $|\Gamma| = 0$ is trivial since $A_1, \cdots, A_m$ and $B_1, \cdots, B_n$ are disjoint. Assume the theorem for $|\Gamma| < k$ $(k > 0)$. We consider the case $|\Gamma| = k$. Let $\Delta = \{M_1, \cdots, M_p\}$. Then we can write $M \equiv C[M_1, \cdots, M_p]$, $N \equiv C[N_1, \cdots, N_p]$, $P \equiv C[P_1, \cdots, P_p]$ where $M_i \underset{1}{\twoheadrightarrow} N_i$ and $M_i \underset{2}{\twoheadrightarrow} P_i$ $(i = 1, \cdots, p)$. We will now show that for every M_i, we can obtain Q_i satisfying the diagram in Figure 3.4.

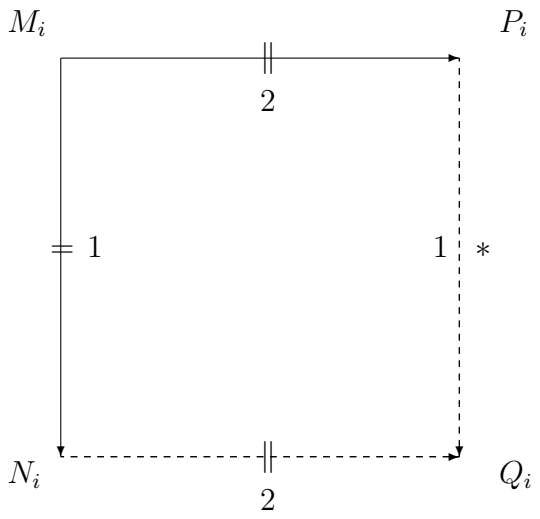


Figure 3.4

There are two cases.

Case 1. $M_i \notin \{B_1, \cdots, B_n\}$.

Then $M_i \xrightarrow[1]{M_i} N_i$ and $M_i \overset{B'_1,\cdots,B'_q}{\underset{2}{\Longrightarrow}} P_i$, where $B'_j \in \{B_1, \cdots, B_n\}$ and $B'_j \subset M_i$ for all B'_j. Let $A \triangleright B \in R_1$, $M_i \equiv A\theta$, and $N_i \equiv B\theta$. If every redex occurrence B'_j of M_i occurs in θ then we can obtain Q_i by Lemma 3.3.2.

Now assume that some B'_j exists which does not occur in θ. Without loss of generality, it may be assumed that B'_1 does not occur in θ. Then there exists $A' \triangleright B' \in R_2$ such that $B'_1 \equiv A'\theta'$. Since $A' \triangleright B'$ overlaps $A \triangleright B$ and $B'_1 \subset M_i$, there is an inside critical pair, say $\langle D, E \rangle$, in $crit_{in}(R_2, R_1)$. Let $M_i \xrightarrow[2]{B'_1} \tilde{M}_i$. Then $\tilde{M}_i \equiv D\theta''$ and $N_i \equiv E\theta''$ for some θ''. From condition (2) of the theorem, $D \underset{1}{\Longrightarrow} E$. Hence we have $\tilde{M}_i \overset{C_1,\cdots,C_r}{\underset{1}{\Longrightarrow}} N_i$. Also, $\tilde{M}_i \overset{B'_2,\cdots,B'_q}{\underset{1}{\Longrightarrow}} P_i$. For the redex occurrences $C_1, \cdots, C_r$ and $B'_2, \cdots, B'_q$ of $\tilde{M}_i$, we take $\Gamma' = \{C_i | \exists B'_j, C_i \subseteq B'_j\} \cup \{B'_i | \exists C_j, B'_i \subseteq C_j\}$. Since $\forall \tilde{B} \in \Gamma'\ \exists B'_j (2 \leq j \leq q),\ \tilde{B} \subseteq B'_j$, we can easily show that $|\Gamma'| \leq \sum_{j=2}^{q} |B'_j|$. Thus $|\Gamma'| \leq \sum_{j=2}^{q} |B'_j| < \sum_{j=1}^{q} |B'_j| \leq |\Gamma|$. Using the induction hypothesis, we obtain the diagram in Figure 3.5.

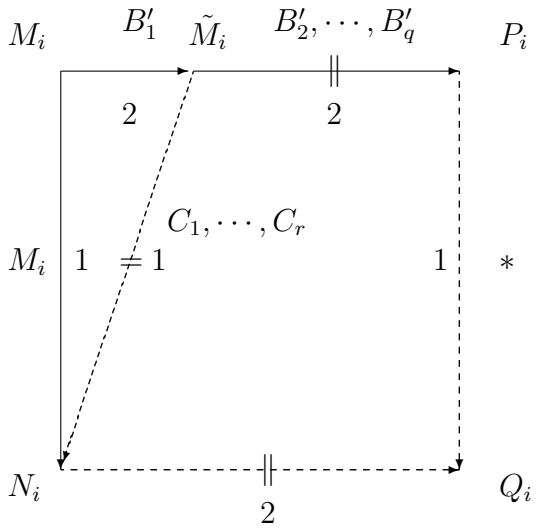


Figure 3.5

Case 2. $M_i \in \{B_1, \cdots, B_n\}$.

Then $M_i \underset{1}{\overset{A'_1,\cdots,A'_q}{\twoheadrightarrow}} N_i$ and $M_i \underset{2}{\overset{M_i}{\to}} P_i$, where $A'_j \in \{A_1, \cdots, A_m\}$ and $A'_j \subseteq M_i$ for all A'_j. Let $A \triangleright B \in R_2$, $M_i \equiv A\theta$, and $P_i \equiv B\theta$. If every redex occurrence A'_j of M_i occurs in θ then we can obtain Q_i by Lemma 3.3.2.

Now assume that some A'_j exists which does not occur in θ. It may be assumed that A'_1 does not occur in θ for the same reason as in case (1). Then there exists $A' \triangleright B' \in R_1$ such that $A'_1 \equiv A'\theta'$. Since $A' \triangleright B'$ overlaps $A \triangleright B$ and $A'_1 \subseteq M_i$, we can obtain a critical pair, say $\langle D, E \rangle$, in $crit(R_1, R_2)$ from this overlapping. Let $M_i \underset{1}{\overset{A'_1}{\to}} \tilde{M}_i$. Then $P_i \equiv E\theta''$ and $\tilde{M}_i \equiv D\theta''$ for some θ''. From condition (1) of the theorem, there is some S such that $D \underset{2}{\twoheadrightarrow} S$ and $D \underset{1}{\overset{*}{\to}} S$. Take $\tilde{P}_i \equiv S\theta''$. Then we have $\tilde{M}_i \underset{2}{\overset{C_1,\cdots,C_r}{\twoheadrightarrow}} \tilde{P}_i$ and $P_i \underset{1}{\overset{*}{\to}} \tilde{P}_i$. Also, $\tilde{M}_i \underset{1}{\overset{A'_2,\cdots,A'_q}{\twoheadrightarrow}} N_i$. For the redex occurrences $A'_2, \cdots, A'_q$ and $C_1, \cdots, C_r$ of $\tilde{M}_i$, we take Γ' in the same way as in case (1); it can be proven that $|\Gamma'| < |\Gamma|$. Using the induction hypothesis, we obtain the diagram in Figure 3.6.

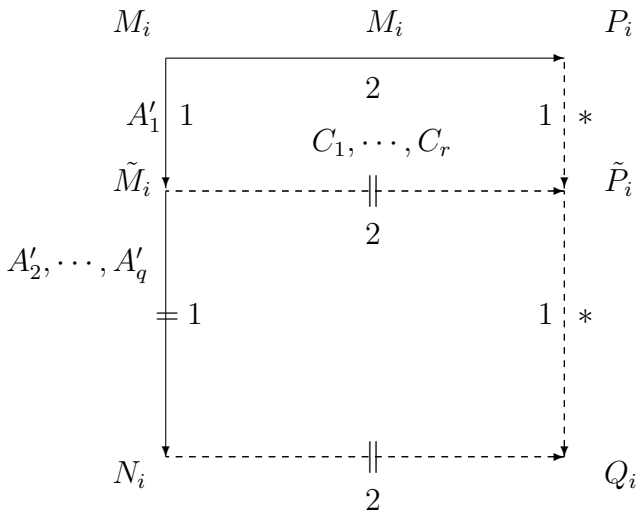


Figure 3.6

Take $Q \equiv C[Q_1, \cdots, Q_p]$. Then it follows that $N \underset{2}{\twoheadrightarrow} Q$ and $P \underset{1}{\overset{*}{\rightarrow}} Q$. □

The following corollary is given in [79, 88].

3.3.4. Corollary. Let left-linear term rewriting systems R_1 and R_2 be nonoverlapping with each other. Then R_1 commutes with R_2.

Proof. It is obvious from Theorem 3.3.3. □

3.3.5. Example. Consider the left-linear term rewriting systems R_1 and R_2:

$$R_1 \quad \begin{cases} f(x) \triangleright h(f(x)) \\ g(x) \triangleright h(g(x)) \end{cases}$$

$$R_2 \quad \begin{cases} f(x) \triangleright g(x) \\ h(f(x)) \triangleright h(g(x)) \end{cases}$$

Then $crit(R_1, R_2) = \{\langle h(f(x)), g(x)\rangle, \langle h(h(f(x))), h(g(x))\rangle\}$ and $crit_{in}(R_2, R_1) = \phi$. It can be shown that $h(f(x)) \underset{2}{\rightarrow} h(g(x))$ and $g(x) \underset{1}{\rightarrow} h(g(x))$ for the critical pair $\langle h(f(x)), g(x)\rangle$, and that $h(h(f(x))) \underset{2}{\rightarrow} h(h(g(x)))$ and $h(g(x)) \underset{1}{\rightarrow} h(h(g(x)))$ for the critical pair $\langle h(h(f(x))), h(g(x))\rangle$. By applying Theorem 3.3.3, it follows that $COM(R_1, R_2)$, i.e., R_1 commutes with R_2.

Let $R = R_1 \cup R_2$. It can easily be shown that $CR(R_1)$ by Proposition 1.4.9 (Rosen) and $CR(R_2)$ by Proposition 1.4.8 (Knuth-Bendix). Thus $CR(R)$ can be obtained from Proposition 1.3.6 (Commutative Union Theorem). Note that non of Propositions 1.4.8, 1.4.9, or 1.4.11 (Huet) can be directly applied to R. □

3.3.6. Example. Consider the left-linear term rewriting systems R_1 and R_2:

$$R_1 \quad \begin{cases} f(x) \triangleright g(f(x)) \\ h(x) \triangleright p(h(x)) \end{cases}$$

$$R_2 \quad \begin{cases} f(x) \triangleright h(f(x)) \\ g(x) \triangleright p(p(h(x))) \end{cases}$$

Then $crit(R_1, R_2) = \{\langle g(f(x)), h(f(x))\rangle\}$ and $crit_{in}(R_2, R_1) = \phi$. It can be shown that $g(f(x)) \underset{2}{\rightarrow} p(p(h(f(x))))$ and $h(f(x)) \underset{1}{\rightarrow} p(h(f(x))) \underset{1}{\rightarrow} p(p(h(f(x))))$ for the critical pair $\langle g(f(x)), h(f(x))\rangle$; by applying Theorem 3.3.3, it follows that R_1 commutes with R_2.

Let $R = R_1 \cup R_2$. We can easily show $CR(R_1)$ and $CR(R_2)$ by Proposition 1.4.9 (Rosen). Thus $CR(R)$ can be obtained from Proposition 1.3.6 (Commutative Union Theorem). It is obvious that non of Propositions 1.4.8, 1.4.9, or 1.4.11 can be

directly applied to R. □

Since self-commuting $COM(R,R)$ and the Church-Rosser property $CR(R)$ are equivalent, we can obtain the following sufficient condition for the Church-Rosser property from Theorem 3.3.3.

3.3.7. Corollary. Let R be a left-linear term rewriting system. Then R has the Church-Rosser property if:

(1) $\forall \langle P, Q\rangle \in crit_{out}(R)\ \exists S\ [P \twoheadrightarrow\!\!\!\!\!\!+\ S \wedge Q \xrightarrow{*} S]$,

(2) $\forall \langle Q, P\rangle \in crit_{in}(R)\ \ [Q \twoheadrightarrow\!\!\!\!\!\!+\ P]$.

Proof. Take $R = R_1 = R_2$. Since $crit(R) = crit_{out}(R) \cup crit_{in}(R)$, we can replace condition (1) of Theorem 3.3.3 with condition (1) of the corollary. Hence the corollary holds. □

Note that Proposition 1.4.11 (Huet) gives a particular case of Corollary 3.3.7.

3.3.8. Example. Consider the left-linear term rewriting system R:

$$R \quad \begin{cases} p(x) \triangleright q(x) \\ p(x) \triangleright r(x) \\ q(x) \triangleright s(p(x)) \\ r(x) \triangleright s(p(x)) \\ s(x) \triangleright f(p(x)) \end{cases}$$

Then $crit_{out}(R) = \{\langle q(x), r(x)\rangle, \langle r(x), q(x)\rangle\}$ and $crit_{in}(R) = \phi$. Since $q(x) \to s(p(x))$ and $r(x) \to s(p(x))$, we can apply Corollary 3.3.7. Thus it is obtained that R has

the Church-Rosser property. Note that the Church-Rosser property of R cannot be proven by applying Proposition 1.4.8, 1.4.9, or 1.4.11. □

3.4. Conclusion

In this chapter we have proposed a new sufficient condition to prove commutativity of left-linear term rewriting systems [98, 92], by extending the critical pair concept to overlapping rewriting rules. The result has extended the sufficent condition for commutativity presented in Raoult and Vuillemin [79] and Sugiyama, Taniguchi and Kasami [88], which does not allow overlapping between two systems. Jouannaud and Kirchner [44] and Jouannaud and Munoz [46] gave the sufficent condition for commutativity of overlapping systems, but they discussed it under the stronger assumption that one of the systems is E-termination. On the other hand, we do not assume termination of systems. We have shown that our result can be applied to proving the Church-Rosser property of left-linear term rewriting systems to which the sufficient conditions proposed by Knuth and Bendix [59], Rosen [81], and Huet [37] cannot directly apply. The proposed result offers a useful means to analyze a complex term rewriting system as the union of its simpler parts.

4. How to Prove Equivalence of Term Rewriting Systems without Induction

In this chapter a simple method, based on the Church-Rosser property and reachability, is proposed for proving the equivalence in a restricted domain of two given term rewriting systems without the explicit use of induction; this proof usually requires some kind of induction. The method is applied to proving the inductive theorems and to deriving a new term rewriting system from a given system by using the equivalence transformation rules. The result is a general extension of the *inductionless induction* methods developed by Musser, Goguen, Huet and Hullot.

4.1. Introduction

We consider how to prove the equivalence in a restricted domain of two term rewriting systems [22, 39, 58, 27] without induction. The equivalence in a restricted domain means that the equational relation (or the transitive reflexive closure) generated by the reduction relation of one system is equal in the restricted domain to that of the other system.

We first explain the concept of the equivalence in a restricted domain through simple examples.

N is the set of natural numbers represented by $0, s(0), s(s(0)), \ldots$. Consider the term rewriting system R_1 computing the addition on the set **N**:

$$R_1 \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y). \end{cases}$$

By adding the associative law to R_1, we can obtain another system R_2 computing the same function:

$$R_2 \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \\ x + (y + z) \triangleright (x + y) + z. \end{cases}$$

Then, R_2 can reduce $(M + N) + P$ and $M + (N + P)$ to the same normal form for any terms M, N, P, while R_1 cannot generally do so unless M, N and P are ground terms. The equivalence of R_1 and R_2 must be regarded as the equivalence in the set of ground terms (i.e., in the initial model) of two systems. Thus we can observe the equivalence in a restricted domain of two term rewriting systems R_1 and R_2.

We show another example. Consider the following term rewriting systems $\tilde{R}_1$ and $\tilde{R}_2$ computing the *double* function $d(n) = 2 * n$:

$$\tilde{R}_1 \quad \begin{cases} d(0) \triangleright 0 \\ d(s(x)) \triangleright s(s(d(x))) \end{cases}$$

$$\tilde{R}_2 \quad \begin{cases} d(x) \triangleright \mathit{if}(x, 0, s(s(d(x - s(0))))) \\ \mathit{if}(0, y, z) \triangleright y \\ \mathit{if}(s(x), y, z) \triangleright z \\ x - 0 \triangleright x \\ s(x) - s(y) \triangleright x - y. \end{cases}$$

The term rewriting system $\tilde{R}_1$ has no infinite reduction sequence. On the other hand, $\tilde{R}_2$ has an infinite reduction sequence starting with the term $d(M)$ for any term M, since the first rewriting rule in $\tilde{R}_2$ can be infinitely applied to the function symbol d. Moreover, $\tilde{R}_1$ has no rewriting rules for the functions *if* and $-$. Thus, $\tilde{R}_1$ and $\tilde{R}_2$ generally produce different reduction sequences, although they can reduce the term $d(M)$ to the same result if M can be reduced to a natural number in $\mathbf{N}$. Therefore, the equivalence of $\tilde{R}_1$ and $\tilde{R}_2$ must be regarded as that in the set of ground terms represented by only the function symbols d, s and 0. Thus, we can say that $\tilde{R}_1$ and $\tilde{R}_2$ are equivalent in the restricted domain.

The concept of equivalence in a restricted domain of term rewriting systems frequently appears in computer science: automated theorem proving, semantics of functional programs, program transformation, verification of programs, and specification of abstract data types. However, this equivalence cannot generally be proved by mere equational reasoning; some kind of induction on the domain structure is necessary.

This chapter presents a new simple method for proving the equivalence in a restricted domain of two term rewriting systems without the explicit use of induction. Our approach to this problem was inspired by the *inductionless induction* methods developed by Musser [69], Goguen [31], Huet and Hullot [38], and others [26, 45, 50, 53, 60, 75, 76, 89]. We generalize the *inductionless induction* methods within a general framework of abstract reduction systems. Some limitations of the *inductionless induction* methods are removed: in particular, the strongly nor-

malizing restriction. The *sufficient completeness* limitation is replaced with a more general concept of *reachability*.

The key idea behind our method is that the equivalence in a restricted domain can be easily proved by the Church-Rosser property and reachability. We first explain this idea in an abstract framework. In Section 4.2, we propose simple sufficient conditions for the equivalence in a restricted domain of two given abstract reduction systems. Our results are carefully partitioned between abstract properties depending solely on the reduction relation and properties depending on the term structure. In Section 4.3, we offer some examples of how to prove the equivalence for term rewriting systems by using the abstract results. In Section 4.4, we explain how our method relates to the *inductionless induction* methods developed by Musser [69], Goguen [31], Huet and Hullot [38]. Based on our framework, we demonstrate that the *inductionless induction* methods can work under a very weak assumption. Finally, in Section 4.5, we propose an equivalence transformation technique for term rewriting systems. Furthermore, we discuss some examples confirming that the extended *inductionless induction* concept is a useful tool for proving correctness of program transformations proposed by Burstall and Darlington [9].

4.2. Equivalence of Abstract Reduction Systems

Let $R_1 = \langle A, \underset{1}{\rightarrow} \rangle$ and $R_2 = \langle A, \underset{2}{\rightarrow} \rangle$ be two abstract reduction systems having the same object set A, and let $\overset{*}{\underset{i}{\rightarrow}}$, $\underset{i}{=}$ and NF_i (i=1,2) be the transitive reflexive closure, the equivalence relation and the set of normal forms in R_i respectively. Note that $\overset{*}{\rightarrow}$ and $=$ are subsets of $A \times A$; for example, $\underset{1}{=} \subseteq \underset{2}{=}$ means that $\forall x, y \in A[x \underset{1}{=} y \Rightarrow x \underset{2}{=} y]$.

4.2.1. Definition. Let A' and A'' be any nonempty subsets of the object set A. Let $\underset{i}{\sim}$ $(i = 1, 2)$ be any two binary relations on A. Then we give the following definitions:

(1) $\underset{1}{\sim} = \underset{2}{\sim}$ in $A' \times A''$ iff $\forall x \in A' \forall y \in A''[x \underset{1}{\sim} y \iff x \underset{2}{\sim} y]$, and $\underset{1}{\sim} \subseteq \underset{2}{\sim}$ in $A' \times A''$ iff $\forall x \in A' \forall y \in A''[x \underset{1}{\sim} y \Rightarrow x \underset{2}{\sim} y]$. We write these relations as $\underset{1}{\sim} = \underset{2}{\sim}$ in A' and $\underset{1}{\sim} \subseteq \underset{2}{\sim}$ in A' respectively if $A' = A''$.

(2) A'' is reachable from A' under $\underset{1}{\sim}$ iff $\forall x \in A' \exists y \in A''[x \underset{1}{\sim} y]$.

(3) A' is closed under $\underset{1}{\sim}$ iff $\forall x \in A' \forall y \in A[x \underset{1}{\sim} y \Rightarrow y \in A']$.

We first show sufficient conditions for $\underset{1}{=} = \underset{2}{=}$ in A'.

4.2.2. Lemma. Let R_1 and R_2 satisfy the following conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $\underset{1}{=} = \underset{2}{=}$ in A'',

(3) A'' is reachable from A' under $\underset{1}{=}$.

Then $\underset{1}{=} = \underset{2}{=}$ in A'.

Proof. Prove $\forall x, y \in A'[x \underset{1}{=} y \iff x \underset{2}{=} y]$. $\Rightarrow$ is trivial from condition (1), hence, we will show $\Leftarrow$. Assume $x \underset{2}{=} y$, where $x, y \in A'$. By using condition (3), there are some elements $z, w \in A''$ such that $x \underset{1}{=} z$ and $y \underset{1}{=} w$. Since $x \underset{2}{=} z$ and $y \underset{2}{=} w$ are obtained from condition (1) , $z \underset{2}{=} w$ can be derived from $z \underset{2}{=} x \underset{2}{=} y \underset{2}{=} w$. From condition (2), $z \underset{1}{=} w$ holds. Therefore $x \underset{1}{=} y$ from $x \underset{1}{=} z \underset{1}{=} w \underset{1}{=} y$. □

If R_2 has the Church-Rosser property, we can modify condition (2) of Lemma 4.2.2 as follows.

4.2.3. Theorem. Assume the following conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $CR(R_2)$, $\underset{2}{\overset{*}{\rightarrow}} \subseteq \underset{1}{\overset{*}{\rightarrow}}$ in A'', and A'' is closed under $\underset{2}{\overset{*}{\rightarrow}}$,

(3) A'' is reachable from A' under $\underset{1}{=}$.

Then $\underset{1}{=} = \underset{2}{=}$ in A'.

Proof. Prove condition (2) of Lemma 4.2.2, i.e., $\forall x, y \in A''[x \underset{1}{=} y \iff x \underset{2}{=} y]$. $\Rightarrow$ is trivial from condition (1), hence we will prove $\Leftarrow$. Assume $x \underset{2}{=} y$, where $x, y \in A''$. From $CR(R_2)$, the closed property of A'' under $\underset{2}{\overset{*}{\rightarrow}}$, and Proposition 1.3.4(i); there exists some $z \in A''$ such that $x \underset{2}{\overset{*}{\rightarrow}} z$ and $y \underset{2}{\overset{*}{\rightarrow}} z$. By using $\underset{2}{\overset{*}{\rightarrow}} \subseteq \underset{1}{\overset{*}{\rightarrow}}$ in A'', $x \underset{1}{\overset{*}{\rightarrow}} z$ and $y \underset{1}{\overset{*}{\rightarrow}} z$ can be derived. Therefore $x \underset{1}{=} y$. □

4.2.4. Theorem. Assume the following conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $CR(R_2)$ and $A'' \subseteq NF_2$,

(3) A'' is reachable from A' under $\underset{1}{=}$.

Then $\underset{1}{=} = \underset{2}{=}$ in A'.

Proof. Prove condition (2) of Lemma 4.2.2, i.e., $\forall x, y \in A''[x \underset{1}{=} y \iff x \underset{2}{=} y]$. $\Rightarrow$ is trivial. $\Leftarrow$: By using condition (2) of this theorem and Proposition 1.3.4(ii), $x \underset{2}{=} y \Rightarrow x \equiv y$ for any $x, y \in A''$. Therefore $x \underset{1}{=} y$. □

4.2.5. Corollary. Assume the conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $CR(R_2)$ and $NF_1 = NF_2$,

(3) $WN(R_1)$.

Then $\underset{1}{=} = \underset{2}{=}$ is obtained.

Proof. Set $A' = A$ and $A'' = NF_1 = NF_2$ in Theorem 4.2.4. □

Next, we consider sufficient conditions for $\underset{1}{\overset{*}{\to}} = \underset{2}{\overset{*}{\to}}$ in $A' \times A''$.

4.2.6. Theorem. Assume the following conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $CR(R_2)$ and $A'' \subseteq NF_2$,

(3) A'' is reachable from A' under $\underset{1}{\overset{*}{\to}}$.

Then $\underset{1}{\overset{*}{\to}} = \underset{2}{\overset{*}{\to}}$ in $A' \times A''$.

Proof. Prove $\forall x \in A' \forall y \in A''[x \underset{1}{\overset{*}{\to}} y \iff x \underset{2}{\overset{*}{\to}} y]$. $\Rightarrow$: Let $x \underset{1}{\overset{*}{\to}} y$. Then $x \underset{2}{=} y$ from condition (1). Thus $x \underset{2}{\overset{*}{\to}} y$ is obtained from condition (2) and Proposition 1.3.4(iii). $\Leftarrow$: Let $x \underset{2}{\overset{*}{\to}} y$. Then, from condition (3), there exists some $z \in A''$ such that $x \underset{1}{\overset{*}{\to}} z$. By condition (1), $x \underset{2}{=} z$; hence, $y \underset{2}{=} z$ can be derived from $y \underset{2}{=} x \underset{2}{=} z$. Thus, $y \equiv z$ is obtained from condition (2) and Proposition 1.3.4(ii). Therefore $x \underset{1}{\overset{*}{\to}} y$. □

4.2.7. Corollary. Assume the conditions:

(1) $\underset{1}{=} \subseteq \underset{2}{=}$,

(2) $CR(R_2)$ and $NF_1 = NF_2$,

(3) $WN(R_1)$.

Then $\underset{1}{\overset{*}{\to}} = \underset{2}{\overset{*}{\to}}$ in $A \times NF_1$.

Proof. Set $A' = A$ and $A'' = NF_1 = NF_2$ in Theorem 4.2.6. □

In the following sections, we will explain how to apply the above abstract results to term rewriting systems that are reduction systems having a term set as the object set. However, note that the above abstract results can be applied not only to term rewriting systems, but also to various reduction systems.

4.3. Examples of Equivalent Systems

We now illustrate how to prove the equivalence in a restricted domain of two term rewriting systems R_1 and R_2 by using Theorems 4.2.3, 4.2.4, and 4.2.6. We will omit here all proofs of reachability in the following examples. For the proofs of reachability for the examples in this and the following sections, see Appendix B.

4.3.1. Example. Let $F' = \{+, s, 0\}$ and $F'' = \{s, 0\}$. Consider the term rewriting systems R_1 and R_2 computing the addition on the set $\mathbf{N}$:

$$R_1 \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

$$R_2 \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \\ x + (y + z) \triangleright (x + y) + z \end{cases}$$

We will prove that $\underset{1}{=} = \underset{2}{=}$ in $T(F')$ by using Theorem 4.2.4. Let $A' = T(F')$, $A'' = T(F'')$ in Theorem 4.2.4. We need to verify conditions (1), (2), (3) of Theorem 4.2.4 for R_1 and R_2. Since $\underset{1}{\triangleright} \subseteq \underset{2}{\triangleright}$, condition (1), i.e., $\underset{1}{=} \subseteq \underset{2}{=}$, is obvious. By using $SN(R_2)$ and Proposition 1.4.8, $CR(R_2)$ is obtained. Condition (2) holds, since $T(F'') \subseteq NF_2$. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. Therefore, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$.

It is also possible to prove $\underset{1}{\overset{*}{\rightarrow}} = \underset{2}{\overset{*}{\rightarrow}}$ in $T(F') \times T(F'')$ by using Theorem 4.2.6. □

The next example shows that our result can be easily applied to the reduction systems having the relation between constructors such as $s^3(x) = x$.

4.3.2. Example. Let $F' = \{+, s, 0\}$ and $F'' = \{s, 0\}$. Consider the term rewriting systems R_1 and R_2 computing the addition on Z_3:

$$R_1 \quad \begin{cases} s(s(s(x))) \triangleright x \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

$$R_2 \quad \begin{cases} s(s(s(x))) \triangleright x \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \\ x + (y + z) \triangleright (x + y) + z \end{cases}$$

We will prove that $\underset{1}{=} = \underset{2}{=}$ in $T(F')$ by using Theorem 4.2.3. Let $A' = T(F')$, $A'' = T(F'')$. We need to verify conditions (1), (2), (3) of Theorem 4.2.3 for R_1 and R_2. Condition (1), i.e., $\underset{1}{=} \subseteq \underset{2}{=}$, is obvious. By using $SN(R_2)$ and Proposition 1.4.8, $CR(R_2)$ is obtained. Condition (2) holds since $\underset{1}{\rightarrow} = \underset{2}{\rightarrow}$ in $T(F'')$, and since $T(F'')$ is closed under $\underset{2}{\rightarrow}$. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. Therefore, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$.

Note that it is also possible to prove $\underset{1}{=} = \underset{2}{=}$ in $T(F')$ by letting $A'' = \{0, s(0), s(s(0))\}$ and by using Theorem 4.2.4. □

We next show an example in which R_2 does not have the strongly normalizing property.

4.3.3. Example. Consider the following term rewriting systems R_1 and R_2 computing the *double* function $d(n) = 2 * n$:

$$R_1 \quad \begin{cases} d(0) \triangleright 0 \\ d(s(x)) \triangleright s(s(d(x))) \end{cases}$$

$$R_2 \quad \begin{cases} d(x) \triangleright \mathit{if}(x, 0, s(s(d(x - s(0))))) \\ \mathit{if}(0, y, z) \triangleright y \\ \mathit{if}(s(x), y, z) \triangleright z \\ x - 0 \triangleright x \\ s(x) - s(y) \triangleright x - y \end{cases}$$

The term rewriting system R_2 does not have the strongly normalizing property, since the first rewriting rule in R_2 can be applied infinitely to the function symbol d.

Let $F' = \{d, s, 0\}$ and $F'' = \{s, 0\}$. We will show that the function d of R_1 equals that of R_2 in the restricted domain $T(F')$, that is, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$. For this purpose, Theorem 4.2.4 is used. Let $A' = T(F')$ and $A'' = T(F'')$. We must verify conditions (1), (2), (3) of Theorem 4.2.4. Since $d(0) \underset{2}{=} 0$ and $d(s(x)) \underset{2}{=} s(s(d(x)))$, condition (1), i.e., $\underset{1}{=} \subseteq \underset{2}{=}$, is obtained. It is obvious that R_2 is left-linear and nonoverlapping. Hence, by using Proposition 1.4.9, R_2 has the Church-Rosser property. Since some function symbol not in F'' appears in the left-hand side of any rewriting rule in R_2, we can obtain $T(F'') \subseteq NF_2$. Thus, condition (2) holds. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. Therefore, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$ holds.

Note that $T(F'')$ is also reachable from $T(F')$ under $\underset{1}{\overset{*}{\rightarrow}}$. Hence, by Theorem 4.2.6, we can prove $\underset{1}{\overset{*}{\rightarrow}} = \underset{2}{\overset{*}{\rightarrow}}$ in $T(F') \times T(F'')$ in the same way as the above proof. □

Remark. Reachability from $T(F)$ to $T(C)$ under $=$ has been called sufficient completeness [14, 33, 45, 50, 51, 53, 60, 70, 75] where C is the set of constructors. Clearly, sufficient completeness is a very particular case of reachability.

It is known that sufficient completeness is undecidable in general [33]. The decidability of sufficient completeness under certain conditions is discussed in [14, 45, 50, 51, 53, 60, 70, 75]. Hence, reachability is also undecidable in general, and the test of reachability under certain conditions has the same difficulty as that of sufficient completeness.

Remark. In the above examples, it is sufficient to consider the term rewriting systems on the set $T(F)$ of ground terms. Hence, it is not necessary to verify the Church-Rosser property on $T(F, V)$ but only on $T(F)$ (i.e., *ground confluence* [26, 77]).

4.4. Inductionless Induction

By using Theorem 4.2.4, an equation whose proof usually requires induction on some data structure can be proved without the explicit use of induction. In this section, we will explain how to prove an equation with the *inductionless induction* method [26, 31, 38, 45, 50, 53, 60, 69, 75] based on a very general framework. The framework makes it clear that the *inductionless induction* methods can work under a very weak assumption: the Church-Rosser property and reachability.

Let R_1 be a term rewriting system on $T(F, V)$ (or on $T(F)$). For a term set T, let $M \underset{1}{=} N$ in T denote $\forall\theta[M\theta, N\theta \in T \Rightarrow M\theta \underset{1}{=} N\theta]$. Now, for given terms $M, N \in T(F', V)$ such that any variable in N also occurs in M, consider the validity of $M \underset{1}{=} N$ in $T(F')$. Note that this validity cannot, in general, be proved by merely equational reasoning, that is, some kind of induction on $T(F')$ usually becomes necessary [31, 38, 69] . However, we can prove that $M \underset{1}{=} N$ in $T(F')$ by using the following theorem without induction.

4.4.1. Theorem. Let R_1 be a term rewriting system on $T(F, V)$ (or on $T(F)$) with reachability from $T(F')$ to $T(F'')$ under $\underset{1}{=}$. Let $R_2 = R_1 \cup \{M \triangleright N\}$. If R_2 has the Church-Rosser property and $T(F'') \subseteq NF_2$, then $M \underset{1}{=} N$ in $T(F')$.

Proof. It is obvious that R_1 and R_2 satisfy conditions (1), (2) and (3) of Theorem 4.2.4 by letting $A' = T(F')$ and $A'' = T(F'')$. Thus, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$. Since $M \triangleright N \in R_2$, we can show that $M \underset{2}{=} N$ in $T(F')$. Therefore, $M \underset{1}{=} N$ in $T(F')$. □

4.4.2. Example. Consider R_1 defining the *half* function $h(n) = n/2$ and the *double* function $d(n) = 2 * n$:

$$R_1 \quad \begin{cases} h(0) \triangleright 0 \\ h(s(0)) \triangleright 0 \\ h(s(s(x))) \triangleright s(h(x)) \\ d(0) \triangleright 0 \\ d(s(x)) \triangleright s(s(d(x))) \end{cases}$$

Let $F' = \{h, d, s, 0\}$ and $F'' = \{s, 0\}$. Now, let us prove $h(d(x)) \underset{1}{=} x$ in $T(F')$ by using Theorem 4.4.1. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. Take $R_2 = R_1 \cup \{h(d(x)) \triangleright x\}$. Then, $CR(R_2)$ by Proposition 1.4.8. Clearly, $T(F'') \subseteq NF_2$. Therefore, $h(d(x)) \underset{1}{=} x$ in $T(F')$. □

When $R_2 = R_1 \cup \{M \triangleright N\}$ does not satisfy the conditions in Theorem 4.4.1, we may find R_3 instead of R_2 such that $CR(R_3)$, $T(F'') \subseteq NF_3$, and $\underset{2}{=} = \underset{3}{=}$ in $T(F')$.

4.4.3. Corollary. Let R_1 be a term rewriting system on $T(F, V)$ (or on $T(F)$) with reachability from $T(F')$ to $T(F'')$ under $\underset{1}{=}$. Let $R_2 = R_1 \cup \{M \triangleright N\}$. If there exists a term rewriting system R_3 satisfying the Church-Rosser property, $T(F'') \subseteq$

NF_3, and $\underset{2}{=} = \underset{3}{=}$ in $T(F')$; then $M \underset{1}{=} N$ in $T(F')$.

In Corollary 4.4.3 if the term rewriting system R_2 is strongly normalizing, then an effective search for R_3 can be executed by applying the Knuth-Bendix completion algorithm [59] to R_2. Thus, by using a modified Knuth-Bendix completion algorithm, we can prove automatically inductive theorems without the explicit use of induction. Hence, it has been called *inductionless induction*.

The original idea of the *inductionless induction* method was proposed by Musser [69] , and has been extended by Goguen [31] , Huet and Hullot [38] , and others [26, 45, 50, 53, 60, 75, 76, 89]. However, their *inductionless induction* methods have many limitations. In particular, the requirement for the strongly normalizing property of R_2 [26, 31, 38, 69, 89] (or the strongly normalizing property on equivalence classes of terms if there are non-oriented equations such as associative/commutative laws [45, 50, 53, 60, 75]) restricts its application, since most term rewriting systems in which functions are denoted by recursive definitions, such as recursive programs, do not satisfy this property.

On the other hand, Corollary 4.4.3 have clarified that we can treat the *inductionless induction* concept itself apart from the Knuth-Bendix completion algorithm. Thus, in the *inductionless induction* proof technique if we do not use the Knuth-Bendix completion algorithm to find R_3, the strong normalizing limitation can be removed from R_2. We next show an example in which R_2 is not strongly normalizing.

4.4.4. Example.

$$R_1 \quad \begin{cases} d(x) \triangleright \mathit{if}(x, 0, s(s(d(x - s(0))))) \\ h(x) \triangleright \mathit{if}(x, 0, \mathit{if}(x - s(0), 0, s(h(x - s(s(0)))))) \\ \mathit{if}(0, y, z) \triangleright y \\ \mathit{if}(s(x), y, z) \triangleright z \\ x - 0 \triangleright x \\ s(x) - s(y) \triangleright x - y \end{cases}$$

Note that the term rewriting system R_1 does not have the strongly normalizing property, since the first and second rules in R_1 can be infinitely applied to the function symbols d and h respectively.

Let $F' = \{d, h, s, 0\}$ and $F'' = \{s, 0\}$. Now, we show that $h(d(x)) \underset{1}{=} x$ in $T(F')$. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. Take $R_2 = R_1 \cup \{h(d(x)) \triangleright x\}$. R_2 is not strongly normalizing since R_1 is so. To easily show the Church-Rosser property of the term rewriting system obtained by adding the rule $h(d(x)) \triangleright x$, we consider R_3 instead of R_2:

$$R_3 \quad \begin{cases} d(0) \triangleright 0 \\ d(s(x)) \triangleright s(s(d(x))) \\ h(0) \triangleright 0 \\ h(s(0)) \triangleright 0 \\ h(s(s(x))) \triangleright s(h(x)) \\ \mathit{if}(0, y, z) \triangleright y \\ \mathit{if}(s(x), y, z) \triangleright z \\ x - 0 \triangleright x \\ s(x) - s(y) \triangleright x - y \\ h(d(x)) \triangleright x \end{cases}$$

Then, $\underset{2}{=} = \underset{3}{=}$ in $T(F')$ can be proved in the same way as for Example 4.3.3. It is shown from Proposition 1.4.8 that R_3 has the Church-Rosser property. Clearly, $T(F'') \subseteq NF_3$. Hence, R_3 satisfies the conditions in Corollary 4.4.3. Therefore, $h(d(x)) \underset{1}{=} x$ in $T(F')$. □

Note. There is a practical difficulty in directly applying the above method to automated deduction. The basis of the method is an equivalence transformation from R_2 to R_3. However, at this stage we know no automatic transformation method in which the strong normalizing limitation on R_2 is unnecessary. Thus, an automatic transformation method must still be found.

4.5. Equivalence Transformation Technique

In this section, we propose the equivalence transformation rules for term rewriting systems. We show that the equivalence of term rewriting systems, to which it is difficult to apply Theorems 4.2.4 and 4.2.6 directly, can be easily proved by an equivalence transformation technique. The results are effectively applied to proving the correctness of program transformations proposed by Burstall and Darlington [9].

Let $R_0 = \langle T, \underset{0}{\rightarrow} \rangle$ with $\triangleright$ be a left-linear term rewriting system having the Church-Rosser property. Here, T is $T(F, V)$ or $T(F)$. Let F' and F'' be the subsets of F and let $T(F'') \subseteq NF_0$. Let $S_0 = \triangleright$. Now, we give the equivalence transformation rules from $R_n = \langle T, \underset{n}{\rightarrow} \rangle$ $(n \geq 0)$ to $R_{n+1} = \langle T, \underset{n+1}{\rightarrow} \rangle$:

I. **Introduction**: Introduce a new relation by adding a new rewriting rule $P \triangleright Q$ to R_n; where $P \triangleright Q$ is not overlapping with any rule in S_n, and P is linear and has at least one function symbol not in F''. Thus, $R_{n+1} = R_n \cup \{P \triangleright Q\}$. Set $S_{n+1} = S_n \cup \{P \triangleright Q\}$.

A. **Addition**: Add an extra rule $P \triangleright Q$ to R_n, where $P \underset{n}{=} Q$. Thus, $R_{n+1} =$

$R_n \cup \{P \triangleright Q\}$. Set $S_{n+1} = S_n$.

E. **Elimination**: Remove a rule $P \triangleright Q$ from R_n. Thus, $R_{n+1} = R_n - \{P \triangleright Q\}$. Set $S_{n+1} = S_n$.

Remark. The above three rules are a natural extension of the program transformation rules suggested by Burstall and Darlington [9] : *Definition, Instantiation, Unfolding, Folding, Abstraction*, and *Laws*. We can easily show that *Instantiation, Unfolding, Folding*, and *Laws* can be obtained directly from rule A; and *Definition* from rule I. *Abstraction* can be also obtained by combining the above three rules. Hence, their program transformations can be seen as a particular case of our equivalence transformations for term rewriting systems in restricted domains. Indeed, we can give a formal proof to the correctness of the program transformations by the technique developed in this section: see Examples 4.5.4 and 4.5.5.

$R_n \underset{i}{\Rightarrow} R_{n+1}$ shows that R_n is transformed to R_{n+1} by rule i ($i = I, A$, or E). $R_n \Rightarrow R_{n+1}$ shows that R_n is transformed to R_{n+1} by rule I, A, or E. $\overset{*}{\underset{i}{\Rightarrow}}$ and $\overset{*}{\Rightarrow}$ denote the transitive reflexive closure of $\underset{i}{\Rightarrow}$ and $\Rightarrow$, respectively.

4.5.1. Lemma. If $R \underset{i}{\Rightarrow} R' \underset{j}{\Rightarrow} \tilde{R}$ ($i = E$ and $j = I$, $i = E$ and $j = A$, or $i = A$ and $j = I$), then there is some R'' such that $R \underset{j}{\Rightarrow} R'' \underset{i}{\Rightarrow} \tilde{R}$.

Proof. From the definitions of the rules, it is obvious. □

4.5.2. Lemma. Let $R \overset{*}{\Rightarrow} \tilde{R}$. Then, there exists a transformation sequence from R to $\tilde{R}$ such that $R \overset{*}{\underset{I}{\Rightarrow}} R' \overset{*}{\underset{A}{\Rightarrow}} R'' \overset{*}{\underset{E}{\Rightarrow}} \tilde{R}$.

Proof. By using Lemma 4.5.1 repeatedly, we can construct a sequence $R \overset{*}{\underset{I}{\Rightarrow}} R' \overset{*}{\underset{A}{\Rightarrow}}$

$R'' \underset{E}{\overset{*}{\Rightarrow}} \tilde{R}$ from $R \overset{*}{\Rightarrow} \tilde{R}$. □

The following corollary described in Chapter 3 plays an essential role in the proof of Theorem 4.5.3.

Corollary 3.3.4. Let left-linear term rewriting systems R_1 and R_2 be nonoverlapping with each other. Then R_1 commutes with R_2.

4.5.3. Theorem. Let R_0 be a left-linear term rewriting system on $T(F, V)$ (or on $T(F)$) having the Church-Rosser property. Let F' and F'' be the subsets of F and let $T(F'') \subseteq NF_0$. Let $R_0 \overset{*}{\Rightarrow} R_m$, and let $T(F'')$ be reachable from $T(F')$ under $\underset{0}{=}$ and under $\underset{m}{=}$ (resp. under $\underset{0}{\overset{*}{\rightarrow}}$ and under $\underset{m}{\overset{*}{\rightarrow}}$). Then, $\underset{0}{=} = \underset{m}{=}$ in $T(F')$ (resp. $\underset{0}{\overset{*}{\rightarrow}} = \underset{m}{\overset{*}{\rightarrow}}$ in $T(F') \times T(F'')$).

Proof. We prove here only $\underset{0}{=} = \underset{m}{=}$ in $T(F')$, since the proof of $\underset{0}{\overset{*}{\rightarrow}} = \underset{m}{\overset{*}{\rightarrow}}$ in $T(F') \times T(F'')$ can be obtained in an analogous way. From Lemma 4.5.2, we may assume that $R_0 \underset{I}{\overset{*}{\Rightarrow}} R_p \underset{A}{\overset{*}{\Rightarrow}} R_q \underset{E}{\overset{*}{\Rightarrow}} R_m$. To prove the theorem we will show that $\underset{0}{=} = \underset{p}{=}$ in $T(F')$ and $\underset{p}{=} = \underset{m}{=}$ in $T(F')$.

Consider $R_0 \underset{I}{\overset{*}{\Rightarrow}} R_p$. It is clear that $\underset{0}{=} \subseteq \underset{p}{=}$. Let R' be the term rewriting system defined by $S_p - S_0$, i.e., the set of new rules introduced through $R_0 \underset{I}{\overset{*}{\Rightarrow}} R_p$. Then R_p is the union of R_0 and R'. Since R' is left-linear and nonoverlapping, $CR(R')$ can be proved by using Proposition 1.4.9. It is obvious that R_0 and R' are nonoverlapping with each other. Hence, by Corollary 3.3.4 and Proposition 1.3.6 (Commutative Union Theorem), $CR(R_p)$ is obtained. Since the left-hand side of each introduced rule has at least one function symbol not in F'' and since $T(F'') \subseteq NF_0$, we can obtain $T(F'') \subseteq NF_p$. It has been assumed that $T(F'')$ is reachable from $T(F')$ under $\underset{0}{=}$. Hence, by using Theorem 4.2.4, we can obtain $\underset{0}{=} = \underset{p}{=}$ in $T(F')$.

From $R_p \underset{A}{\overset{*}{\Rightarrow}} R_q$, $\underset{p}{=} = \underset{q}{=}$ is trivial.

Now consider $R_q \overset{*}{\underset{E}{\Rightarrow}} R_m$. Since $\underset{m}{=} \subseteq \underset{q}{=}$ and $\underset{q}{=} = \underset{p}{=}$, it is obvious that $\underset{m}{=} \subseteq \underset{p}{=}$. It has been shown that $CR(R_p)$ and $T(F'') \subseteq NF_p$. It has been assumed that $T(F'')$ is reachable from $T(F')$ under $\underset{m}{=}$. Hence, by Theorem 4.2.4 for R_m and R_p, it can be proved that $\underset{p}{=} = \underset{m}{=}$ in $T(F')$.

Therefore, it follows that $\underset{0}{=} = \underset{m}{=}$ in $T(F')$. □

We now explain how to show the equivalence of two term rewriting systems by using the equivalence transformation technique. We take the examples discussed in [9] as examples of program transformations. Thus, the following examples make it clear that the extended *inductionless induction* concept is a useful tool for proving the correctness of the program transformations.

4.5.4. Example (Summation). Consider the following term rewriting systems R_1 and R_2 computing the summation function $f(n) = n + \cdots + 1 + 0$:

$$R_1 \quad \begin{cases} f(0) \triangleright 0 \\ f(s(x)) \triangleright s(x) + f(x) \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

$$R_2 \quad \begin{cases} f(0) \triangleright 0 \\ f(s(x)) \triangleright g(x, s(x)) \\ g(0, y) \triangleright y \\ g(s(x), y) \triangleright g(x, y + s(x)) \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

Let $F' = \{f, +, s, 0\}$ and $F'' = \{s, 0\}$. By using the equivalence transformation rules, we will show that $\underset{1}{=} = \underset{2}{=}$ in $T(F')$. Note that R_2 is in *iterative form* [9]. Hence, this transformation technique can be used to convert *recursive form* into *iterative form* for functional programs.

To transform R_1 to R_2, we first add the associative law for $+$ to R_1: take $R_3 = R_1 \cup \{x + (y + z) \triangleright (x + y) + z\}$. Here, $R_3 \Rightarrow R_1$ by rule E. From Proposition 1.4.8, $CR(R_3)$ is proved. Clearly, $T(F'') \subseteq NF_3$. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$ (and also under $\underset{3}{=}$). By Theorem 4.5.3, $\underset{1}{=} = \underset{3}{=}$ in $T(F')$ is obtained.

Now let us transform R_3 to R_2 by using the transformation rules. Through rule I, we introduce a new function g:

(1) $g(x, y) \triangleright y + f(x)$.

Let $R_4 = R_3 \cup \{(1)\}$, then we can prove

$f(s(x)) \underset{4}{=} g(x, s(x))$,

$g(0, y) \underset{4}{=} y$,

$g(s(x), y) \underset{4}{=} y + f(s(x)) \underset{4}{=} y + (s(x) + f(x)) \underset{4}{=} (y + s(x)) + f(x) \underset{4}{=} g(x, y + s(x))$.

Utilizing rule A, we can obtain $R_5 = R_4 \cup \{(2), (3), (4)\}$:

(2) $f(s(x)) \triangleright g(x, s(x))$,

(3) $g(0, y) \triangleright y$,

(4) $g(s(x), y) \triangleright g(x, y + s(x))$.

Finally, by employing rule E, remove unnecessary rules $x + (y + z) \triangleright (x + y) + z$, $f(s(x)) \triangleright s(x) + f(x)$, and $g(x, y) \triangleright y + f(x)$ from R_5. Thus, we can obtain R_2. $T(F'')$ is reachable from $T(F')$ under $\underset{2}{=}$. Hence, $\underset{3}{=} = \underset{2}{=}$ in $T(F')$ is obtained by Theorem 4.5.3.

Now, we obtain an equivalence transformation sequence from R_1 to R_2: $R_1 \Leftarrow R_3 \Rightarrow R_4 \overset{*}{\Rightarrow} R_5 \overset{*}{\Rightarrow} R_2$. Therefore, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$. It is also possible to prove $\underset{1}{\overset{*}{\rightarrow}} = \underset{2}{\overset{*}{\rightarrow}}$ in $T(F') \times T(F'')$ by analogous transformation. □

4.5.5. Example (Fibonacci). Consider the following term rewriting systems R_1 and R_2 computing the *Fibonacci* function $f(n+2) = f(n+1) + f(n)$ where $f(0) = 1, f(1) = 1$:

$$R_1 \quad \begin{cases} f(0) \triangleright s(0) \\ f(s(0)) \triangleright s(0) \\ f(s(s(x))) \triangleright f(s(x)) + f(x) \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

$$R_2 \quad \begin{cases} f(0) \triangleright s(0) \\ f(s(0)) \triangleright s(0) \\ f(s(s(x))) \triangleright p(h(g(x))) \\ g(0) \triangleright \langle s(0), s(0) \rangle \\ g(s(x)) \triangleright h(g(x)) \\ h(x) \triangleright \langle p(x) + q(x), p(x) \rangle \\ p(\langle x, y \rangle) \triangleright x \\ q(\langle x, y \rangle) \triangleright y \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

Let $F' = \{f, +, s, 0\}$ and $F'' = \{s, 0\}$. By using the equivalence transformation rules, we will show that $\underset{1}{=} = \underset{2}{=}$ in $T(F')$. Note that if we ignore the computation time for the function $+$ as a primitive function, R_2 computes the *Fibonacci* function

in linear time instead of exponential time [9].

From Proposition 1.4.8, $CR(R_1)$ is proved. Clearly, $T(F'') \subseteq NF_1$. $T(F'')$ is reachable from $T(F')$ under $\underset{1}{=}$. To transform R_1 to R_2, we first introduce the pairing $\langle\ ,\ \rangle$ and the projection functions p, q by rule I:

(1) $p(\langle x, y\rangle) \triangleright x$,

(2) $q(\langle x, y\rangle) \triangleright y$.

Through rule I, we introduce a new function g:

(3) $g(x) \triangleright \langle f(s(x)), f(x)\rangle$.

Let $R_3 = R_1 \cup \{(1), (2), (3)\}$. Then we can prove

$g(0) \underset{3}{=} \langle s(0), s(0)\rangle$,

$g(s(x)) \underset{3}{=} \langle f(s(x)) + f(x), f(s(x))\rangle \underset{3}{=} \langle p(g(x)) + q(g(x)), p(g(x))\rangle$.

Hence, we can add the following rules by using rule A:

(4) $g(0) \triangleright \langle s(0), s(0)\rangle$,

(5) $g(s(x)) \triangleright \langle p(g(x)) + q(g(x)), p(g(x))\rangle$.

Here, we introduce a new function h through rule I to avoid computing $g(x)$ three times in the right-hand side of (5),

(6) $h(x) \triangleright \langle p(x) + q(x), p(x)\rangle$.

Let $R_4 = R_3 \cup \{(4), (5), (6)\}$. Then we can prove

$g(s(x)) \underset{4}{=} h(g(x))$,

$f(s(s(x))) \underset{4}{=} p(g(s(x))) \underset{4}{=} p(h(g(x)))$.

By employing rule A, we can obtain $R_5 = R_4 \cup \{(7), (8)\}$:

(7) $g(s(x))) \triangleright h(g(x))$,

(8) $f(s(s(x))) \triangleright p(h(g(x)))$.

Finally, by using rule E, remove unnecessary rules $f(s(s(x))) \triangleright f(s(x)) + f(x)$, $g(x) \triangleright \langle f(s(x)), f(x)\rangle$ and $g(s(x)) \triangleright \langle p(g(x)) + q(g(x)), p(g(x))\rangle$ form R_5. Thus, we can obtain R_2. $T(F'')$ is reachable from $T(F')$ under $\underset{2}{=}$. Hence, $\underset{1}{=} = \underset{2}{=}$ in $T(F')$ is obtained by Theorem 4.5.3. We can also prove $\xrightarrow[1]{*} = \xrightarrow[2]{*}$ in $T(F') \times T(F'')$ by analogous method. □

4.6. Conclusion

In this chapter, we have proposed a new simple method to prove the equivalence in a restricted domain for reduction systems without the explicit use of induction. The key idea is that the equivalence in the restricted domain can be easily tested by using the Church-Rosser property and reachability of reduction systems. Our method has extended the *inductionless induction* methods developed by Musser [69], Goguen [31], Huet and Hullot [38], and others [26, 45, 50, 53, 60, 75, 76, 89] as follows:

(1) The *inductionless induction* methods are based on the framework of term rewriting systems. Our method is essentially based on a more general framework of abstract reduction systems in which we assume only abstract structures. Hence, our method can be applied directly not only to term rewriting systems, but also to various reduction systems: *Thue* systems [5], graph rewriting systems [80], lambda calculus [2], combinatory reduction systems [56], resolution systems [35, 49, 76], and so on.

(2) The *inductionless induction* methods deal with only the inductive equality of two equational theories. On the other hand, our method extends the *inductionless induction* concept to a computational aspect, i.e., *reduction*. The method can deal with the inductive equality of two reduction systems, i.e., the

inductive equality of two *reductions*, in the same way as that of two equational theories. This extension is very important in practice; for instance, we have shown that our results can be effectively applied to proving the correctness of program transformations proposed by Burstall and Darlington [9].

(3) The *inductionless induction* methods have many limitations [38] for term rewriting systems: the Church-Rosser property, the strongly normalizing property, the partition of the function symbols into constructors and non-constructors, and sufficient completeness. These limitations were partially relaxed by several authors [26, 45, 50, 53, 60, 75, 76, 89]; for example, the second restriction can be replaced with the strongly normalizing property on equivalence classes of terms if there are non-oriented equations such as associative/commutative laws [45, 53, 60, 75], and the third restriction can be completely removed [45, 50]. However, our method has made it clear that the *inductionless induction* concept essentially requires only two limitations: the Church-Rosser property and reachability which is an abstract extension of sufficient completeness.

We believe that our method provides a very useful means of proving the equivalence which arises in various formal systems: automated theorem proving, semantics of functional programs, program transformation, program verification, and specification of abstract data types.

5. Membership Conditional Term Rewriting Systems

In this chapter we propose a new type of conditional term rewriting systems: the membership-conditional term rewriting system, in which, each rewriting rule can have membership conditions which restrict the substitution values for the variables occurring in the rule. We study the confluence of membership-conditional term rewriting systems that are *nonterminating* and *nonlinear*. It is shown that a restricted *nonlinear* term rewriting system in which membership conditions satisfy the closure and termination properties is confluent if the system is nonoverlapping.

5.1. Introduction

Many term rewriting systems and their modifications are considered in logic, automated theorem proving, and programming language [3, 39, 48, 54, 58]. A fundamental property of term rewriting systems is the confluence property. A few sufficient criteria for the confluence are well known. However, if a term rewriting system is nonterminating and nonlinear, we know few criteria for the confluence of the system [56, 95].

In this chapter, we study the confluence of membership-conditional term rewriting systems that are nonterminating and nonlinear. In a membership-conditional term rewriting system, the rewriting rule can have membership conditions.

We explain this concept with an example. We first consider a classical term rewriting system R that is nonterminating and nonlinear:

$$R \quad \begin{cases} f(x,x) \triangleright 0 \\ f(g(x),x) \triangleright 1 \\ 2 \triangleright g(2) \end{cases}$$

The diagram in Figure 5.1 illustrates that R is not confluent:

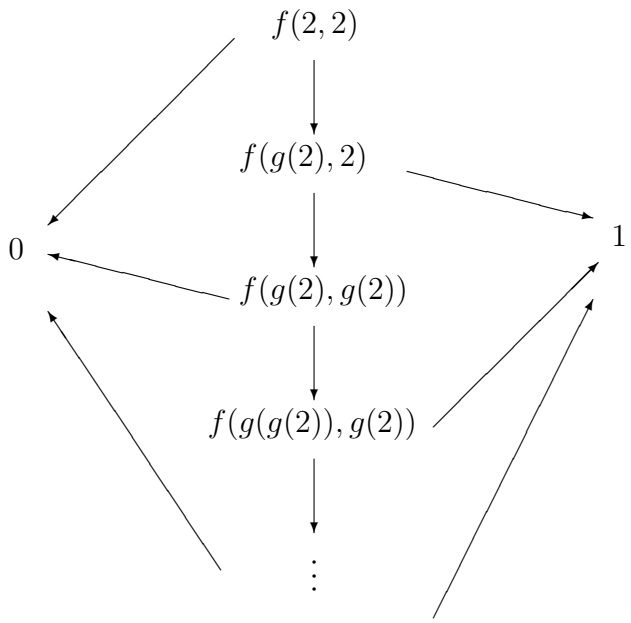


Figure 5.1

Now, let T' be the set of terms containing no constant symbol 2. By adding the membership condition $x \in T'$ to the first and second rules in R, we obtain the membership-conditional term rewriting system R':

$$R' \quad \begin{cases} f(x,x) \triangleright 0 \ \textit{if}\ x \in T' \\ f(g(x),x) \triangleright 1 \ \textit{if}\ x \in T' \\ 2 \triangleright g(2) \end{cases}$$

The membership condition $x \in T'$ restricts the substitution values for variable x; for example, the first rule $f(x,x) \triangleright 0$ *if* $x \in T'$ defines the reduction $f(M,M) \to 0$ only when $M \in T'$. Then, we can prove that R' is confluent (see Example 5.3.9 in Section 5.3), though it is nonterminating and nonlinear. Thus, by adding appropriate membership conditions, nonlinear systems can easily have the confluence property.

Our idea of membership-conditional rewriting was inspired by Church's δ-rule in λ-calculus [2, 56]:

$$\delta_C \quad \begin{cases} \delta M M \triangleright \mathbf{T} & \textit{if } M \textit{ is a closed normal form} \\ \delta M N \triangleright \mathbf{F} & \textit{if } M, N \textit{ are closed normal forms and } M \not\equiv N. \end{cases}$$

It is well known that λ-calculus with δ_C is confluent [2, 56]. However, if λ-calculus has Hindley's δ-rule

$$\delta_H \quad \begin{cases} \delta M M \triangleright M \end{cases}$$

or Staples's δ-rule

$$\delta_S \quad \begin{cases} \delta M M \triangleright \epsilon \end{cases}$$

instead of δ_C, then it is not confluent [2, 56]. Thus, the membership conditions in δ_C (i.e., M, N *must be in the set of closed normal forms*) play an important role for the confluence of λ-calculus with nonlinear rules.

We will extend the idea of membership-conditional rewriting offered in Church's δ-rule to nonlinear term rewriting systems. Section 5.2 introduces the concept of membership-conditional term rewriting systems. In Sections 5.3, we discuss the sufficient criteria for the confluence of membership-conditional term rewriting systems that are nonterminating and nonlinear. We show that a restricted nonlinear system in which the membership conditions satisfy the closure and termination properties is confluent if the system is nonoverlapping.

5.2. Membership-Conditional Rewriting

In this section, we propose membership-conditional term rewriting systems. A membership-conditional term rewriting system R on T is a term rewriting system on T in which the rewriting rule $M_l \triangleright M_r$ can have the membership conditions $x \in T', y \in T'', \cdots, z \in T'''$. Here, $T', T'', \cdots, T'''$ are any subsets of T.

The membership-conditional rewriting rule is denoted by

$$M_l \triangleright M_r \ \textit{if}\ x \in T', y \in T'' \cdots, z \in T'''.$$

The conditions $x \in T', y \in T'' \cdots, z \in T'''$ restrict the substitution's values on the variables $x, y, \cdots, z$ occurring in the rule $M_l \triangleright M_r$. Thus, the rule $M_l \triangleright M_r$ *if* $x \in T', y \in T'' \cdots, z \in T'''$ defines the reduction $M \rightarrow N$ only when $M \equiv C[M_l\theta]$, $N \equiv C[M_r\theta]$ for some $C[\ ]$ and some θ such that $x\theta \in T', y\theta \in T'', \cdots, z\theta \in T'''$.

5.2.1. Example. Let $F = \{+, d, s, 0\}$ and $F' = \{+, s, 0\}$. Consider the membership-conditional term rewriting system R on $T(F, V)$ which computes the addition and the double function $d(n) = n + n$ on the set $\mathbf{N}$ of natural numbers represented by $0, s(0), s(s(0)), \ldots$:

$$R \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \\ d(x) \triangleright x + x \ \textit{if}\ x \in T(F') \end{cases}$$

Then we have the following reduction:

$$d(d(0)) \rightarrow d(0 + 0) \rightarrow (0 + 0) + (0 + 0) \stackrel{*}{\rightarrow} 0.$$

Note that $d(d(0))$ cannot directly contract into $d(0) + d(0)$ with the third rule in R since $d(0) \notin T(F')$. □

5.2.2. Example. Let $F = \{-, s, 0\}$. Consider the membership-conditional

term rewriting system R on $T(F, V)$ computing the subtraction on the set $\mathbf{N}$:

$$R \quad \begin{cases} x - 0 \triangleright x \ \mathit{if}\, x \in NF \\ s(x) - s(y) \triangleright x - y \ \mathit{if}\, x, y \in NF \\ x - x \triangleright 0 \ \mathit{if}\, x \in NF \end{cases}$$

Then, R contracts only the innermost redex occurrences in a term since the membership conditions prohibit to contract the other redex occurrences. Thus, by using the membership conditions we can explicitly provide the innermost reduction strategy for term rewriting systems. □

Note that we allow any (not necessarily decidable) membership condition $x \in T'$. However, if a membership condition is undecidable, the membership-conditional system R might not be well-defined (i.e., the reduction relation $\rightarrow$ of R cannot be defined). For example, a rewriting rule in which a membership condition restricts the application of itself leads us to the following paradoxical system R:

$$R \quad \left\{ \ f(x) \triangleright 0 \ \mathit{if}\, x \in \{M | f(M) \in NF\} \right.$$

Then, we can show that $f(0)$ is a normal form iff $f(0)$ is not a normal form: a contradiction. Hence R is not well-defined.

As regarding Examples 5.2.1, the membership-conditional system is well-defined since the condition $x \in T(F')$ in the third rule is obviously decidable. From the following lemma, we can show that the systems in Examples 5.2.2 and 5.3.7 (in Section 5.3) are also well-defined.

Lemma 5.2.3. Let R be a membership-conditional system in which each condi-

tion has the form $x \in NF$ (where x may be any variable). Then, R is well-defined.

Proof. Consider the claim: $M \in NF$ (i.e., the irreducibility for M) is decidable for any term M. It is clear that the lemma follows from this claim. We will prove the claim by induction on the size $|M|$ of the term M (i.e., the number of the symbols occurring in M). The case $|M| = 1$ is trivial since M is a variable or a constant. Assume the lemma for $|M| < k$. Then, we must show the lemma for the case $|M| = k$. It is decidable whether M has a redex as a proper subterm, say P, by $|P| < |M|$ and the induction hypothesis. We will show that it is also decidable whether M is a redex. Consider a rule $M_l \triangleright M_r$ *if* $x, \cdots, z \in NF$. Then, M is a redex for this rule iff $M \equiv M_l\theta$ and $x\theta, \cdots, z\theta \in NF$ for some θ. By $|x\theta|, \cdots, |z\theta| < |M|$ and the induction hypothesis, we can decide whether M is a redex for the rule. Thus, testing every rule in R, we can decide whether M is a redex. Therefore, the decidability of $M \in NF$ follows. □

In this chapter, we are interested in only well-defined membership-conditional systems. Thus, from here on "a membership-conditional system R" means implicitly that R is well-defined.

Remark. In the membership-conditional system R with undecidable conditions, the rewriting of any term is in general an undecidable problem. Nevertheless, this does not necessarily mean that R is not well-defined; for example, we might indirectly compute the normal forms by using other ways than rewriting.

Remark. A conditional rule $M_l \triangleright M_r$ *if* $P(x)$ [3], where $P(x)$ is some predicate of the variable x, can be translated into a membership-conditional rule $M_l \triangleright M_r$ *if* $x \in T$ where $T = \{N|\ P(N)\}$. Conversely, taking $P(x) \equiv x \in T$, we can also translate a membership-conditional rule $M_l \triangleright M_r$ *if* $x \in T$ into a conditional rule $M_l \triangleright M_r$ *if* $P(x)$. Thus conditional rules of the form

$$M_l \triangleright M_r \ \textit{if}\ P'(x) \wedge P''(y) \wedge \cdots \wedge P'''(z)$$

are essentially equal to membership-conditional rules of the form

$$M_l \triangleright M_r \ \textit{if}\ x \in T', y \in T'' \cdots, z \in T'''.$$

Hence a membership-conditional term rewriting system can be regarded as a conditional term rewriting system in which every condition $P(x, y, \cdots, z)$ can be translated into a condition $P'(x) \wedge P''(y) \wedge \cdots \wedge P'''(z)$ with separated variables.

5.3. Confluence of Restricted Nonlinear Systems

It is well known that if a term rewriting system is terminating, the confluence can be easily proved by Proposition 1.4.8 (Knuth-Bendix). However, if a term rewriting system is nonterminating, it is difficult to prove the confluence of the system. In particular, a system that is nonterminating and nonlinear gives few results to prove the confluence [56, 95].

In this section, we study the confluence of membership-conditional term rewriting systems without assuming the terminating property or the linearity. First we shall consider a membership-conditional system with *normalized* conditions (i.e. each set appearing in conditions must be a set of normal forms).

Let $R = \langle T, \rightarrow \rangle$ be a membership-conditional term rewriting systems and let T' be a subset of the term set T. We say that T' is closed iff
$\forall M \in T' \ \forall N \in T[M \rightarrow N \Rightarrow N \in T']$. We say that T' is terminating iff every $M \in T'$ has no infinite reduction $M \rightarrow\rightarrow\rightarrow \cdots$.

For a term set T' closed and terminating, we can define the normalized term set $T'_{nf} = \{M \downarrow \ \mid M \in T'\}$ where $M \downarrow$ denotes any normal form obtained from M. Note that from the closure and termination properties of T', T'_{nf} is definable and $T'_{nf} \subseteq T'$. Then, the normalized membership-conditional system R_{nf} is defined by replacing each rewriting rule

$M_l \triangleright M_r$ *if* $x \in T', \cdots, z \in T''$ in R with
$M_l \triangleright M_r$ *if* $x \in \psi(T'), \cdots, z \in \psi(T'')$. Here, $\psi(T') = T'_{nf}$ if T' is closed and terminating; otherwise $\psi(T') = T'$. $\underset{nf}{\longrightarrow}$ denotes the reduction relation of R_{nf}. From the closed property, it is trivial that $\underset{nf}{\longrightarrow} \subseteq \rightarrow$. We will show that if R_{nf} is confluent then R is so.

5.3.1. Lemma. Let T' be closed and terminating in R and let $M \in T'$. Then, $M \underset{nf}{\overset{*}{\rightarrow}} M \downarrow$ for some $M \downarrow \in T'_{nf}$.

Proof. Since T' is terminating, R can reduce M into some $M \downarrow$ by rewriting only innermost $\rightarrow$ redex occurrences (i.e., innermost reduction strategy). From the definition of R_{nf}, every innermost $\rightarrow$ redex occurrence is an innermost $\underset{nf}{\rightarrow}$ redex occurrence. Thus, by tracing the innermost reduction $M \overset{*}{\rightarrow} M \downarrow$ by R_{nf}, $M \underset{nf}{\overset{*}{\rightarrow}} M \downarrow$ follows. □

5.3.2. Lemma. We have the diagram in Figure 5.2.

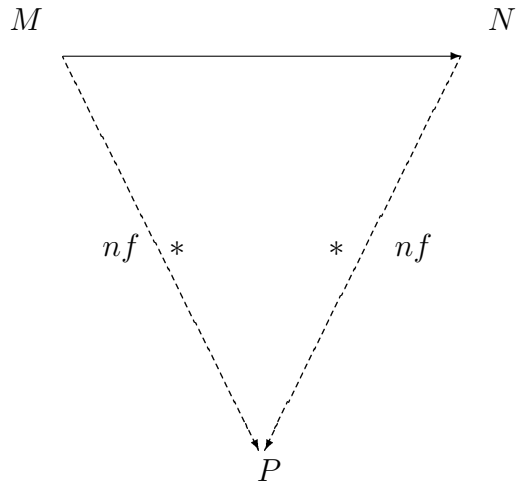


Figure 5.2

Proof. For example, let R have a reduction $M \equiv C[f(A, A, B)] \rightarrow N \equiv C[g(A, B, B)]$ by a rule $f(x, x, y) \triangleright g(x, y, y)$ *if* $x \in T', y \in T''$, and let T' be closed and terminating and T'' be not (i.e. $\psi(T') = T'_{nf}$ and $\psi(T'') = T''$). Then, the nor-

malized rule is $f(x,x,y) \triangleright g(x,y,y)$ *if* $x \in T'_{nf}, y \in T''$. From Lemma 5.3.1, $A \xrightarrow[nf]{*} A\downarrow$ for some $A\downarrow \in T'_{nf}$. Hence, R_{nf} have the reductions
$C[f(A,A,B)] \xrightarrow[nf]{*} C[f(A\downarrow, A\downarrow, B)]$ and $C[g(A,B,B)] \xrightarrow[nf]{*} C[g(A\downarrow, B, B)]$.
By using the normalized rule, $C[f(A\downarrow, A\downarrow, B)] \underset{nf}{\rightarrow} C[g(A\downarrow, B, B)]$. Thus, take $P \equiv C[g(A\downarrow, B, B)]$. It is clear that for any M, N, we can always take some P in the same way as for the above example. □

5.3.3. Lemma. If R_{nf} is confluent then we have the diagram in Figure 5.3.

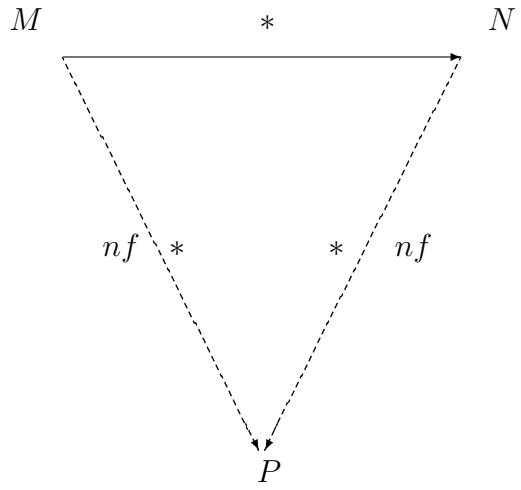


Figure 5.3

Proof. Let $M \xrightarrow{n} N$, where $\xrightarrow{n}$ denotes a reduction of n ($n \geq 0$) steps. Then we prove the lemma by induction on n. The case $n = 0$ is trivial. Assume the claim for $n-1$ ($n > 0$). Let $M \rightarrow M' \xrightarrow{n-1} N$. Then, the diagram in Figure 5.4 can be obtained, where diagram (1) is shown by Lemma 5.3.2, diagram (2) by the induction hypothesis, and diagram (3) by the confluence of R_{nf}. □

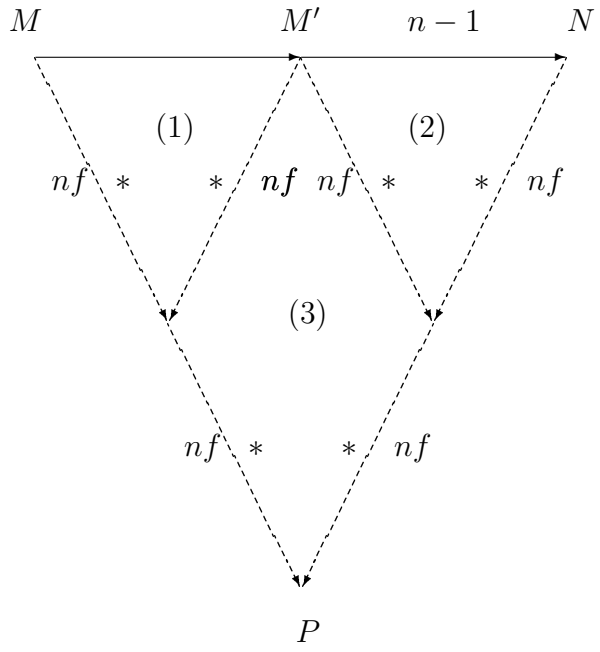


Figure 5.4

5.3.4. Theorem. If R_{nf} is confluent then R is so.

Proof. The diagram in Figure 5.5 can be obtained, proving diagram(1) by Lemma 5.3.3, diagram(2) by the confluence of R_{nf}. From $\underset{nf}{\rightarrow} \subseteq \rightarrow$, the confluence of R follows. □

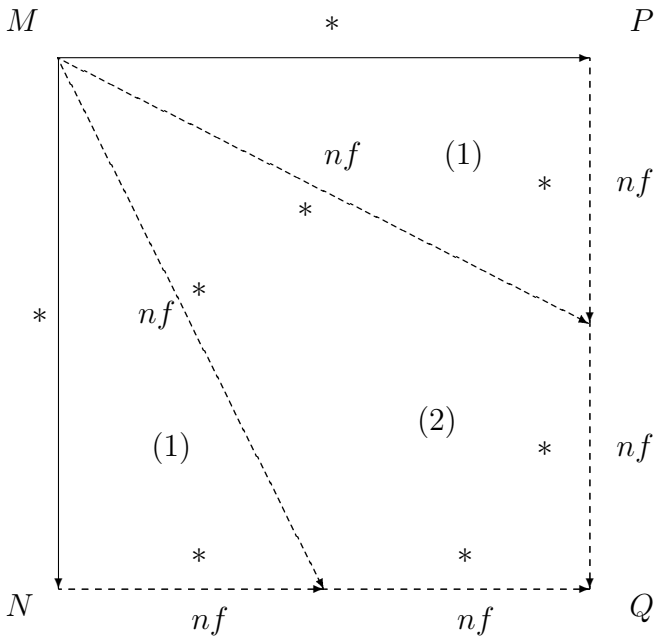


Figure 5.5

Now, we study the confluence of a membership-conditional term rewriting system in which each nonlinear variable in the left-hand side of the rules is restricted with a membership condition. Our key idea to prove the confluence comes from the observation that with appropriate membership conditions, nonlinear systems behave like left-linear systems.

5.3.5. Definition. A restricted nonlinear rule is a membership-conditional rewriting rule in which the nonlinear variables on the left side of the rule must have membership conditions. For the other variables, membership conditions are optional. We say that R is restricted nonlinear iff every rule in R is restricted nonlinear.

For example, the restricted nonlinear rule $f(x, x, y) \triangleright g(x, y, y)$ *if* $x \in T'$ has nonlinear variable x on the left side $f(x, x, y)$. Hence, variable x must have the

membership condition $x \in T'$. However, variable y on the left side is linear, thus, membership condition for y is not necessary.

A classical left-linear term-rewriting system is obviously a restricted nonlinear system, because the left-linear system has only linear variables on the left side of the rewriting rules. Thus, the restricted nonlinear system is a natural extension of the classical left-linear system. Indeed, the sufficient criteria for the confluence of restricted nonlinear systems are very similar to that of the classical left-linear systems.

Overlapping between two conditional rewriting rules can be defined in the same way as for two classical rewriting rules except that the substitution must satisfy the membership conditions in the rules. Then, Proposition 1.4.9 (Rosen) for the confluence of the classical left-linear systems can be extended to the following theorem.

5.3.6. Theorem. Let a membership-conditional term rewriting system R be nonoverlapping and restricted nonlinear. If every term set T' in the membership conditions is a set of normal forms, i.e., $T' \subseteq NF$, then R is confluent.

Proof. Since nonlinear variables on the left side of the rewriting rules must have normal forms as the substitution's values, the nonlinear variables can be ignored when we treat a sufficient criterion for the confluence. Thus, the confluence of R can be easily proved in the same way as for the classical left-linear and nonoverlapping systems, by tracing the proof in [37, 81] of Proposition 1.4.9. □

5.3.7. Example. Consider the membership-conditional term rewriting system R:

$$R \quad \begin{cases} f(x,x) \triangleright 0 \ \textit{if} \ x \in NF \\ f(g(x),x) \triangleright 1 \ \textit{if} \ x \in NF \\ 2 \triangleright g(2) \end{cases}$$

Note that R is nonterminating and nonlinear. Clearly, R satisfies the conditions in Theorem 5.3.6. Thus, R is confluent. □

In Theorem 5.3.6, every set T' in the membership conditions must be a set of normal forms. We are now going to relax this restriction on the membership conditions by Theorem 5.3.4.

5.3.8. Theorem. Let a membership-conditional term rewriting system R be nonoverlapping and restricted nonlinear. If every term set T' in the membership conditions is closed and terminating, then R is confluent.

Proof. From Theorems 5.3.4 and 5.3.6, the theorem follows. □

5.3.9. Example. Let $F' = \{f, g, 0, 1\}$. Consider the membership conditional term rewriting system R:

$$R \quad \begin{cases} f(x,x) \triangleright 0 \ \textit{if} \ x \in T(F',V) \\ f(g(x),x) \triangleright 1 \ \textit{if} \ x \in T(F',V) \\ 2 \triangleright g(2) \end{cases}$$

It is clear that R is nonoverlapping and restricted nonlinear. Since $T(F', V)$ is closed and terminating, from Theorem 5.3.8 it follows that R is confluent. □

5.4. Conclusion

In this chapter, we have proposed a new conditional term rewriting system: the membership-conditional term rewriting system. We have shown the sufficient criteria for the confluence of the system under the restricted nonlinear condition [101, 97]. The work of Kirchner [54] about meta-rules has a closed connection to our work, since a set of meta-rules can be considered as a membership-conditional term rewriting system. It is mentioned in [54] that our result gives a sufficent criterion for the confluence of the non-overlapping meta-rewriting systems.

Many directions for further research come easily to mind. One direction is application to many-sorted systems. Membership-conditional systems can provide a very useful means of constructing hierarchical many-sorted systems [33, 54]. Application to functional programs [104, 105, 40] is another very interesting direction. Membership-conditional systems can explicitly provide reduction strategy, such as innermost reduction. Hence, using this property, we can offer effective computation for functional programs. We believe that further research in these directions will exploit the potential of membership-conditional rewriting techniques.

6. Conclusion

We have investigated several topics for term rewriting systems having the Church-Rosser property. The results in this thesis are summarized as follows:

(1) Modularity has been developed for the direct sum of term rewriting systems. It has been demonstrated that the Church-Rosser property and the left-linear complete property both are modular, but the termination property is not. Moreover, a criterion has been proposed for commutativity of term rewriting systems, which also offers modularity for the union of Church-Rosser systems without the requirement of the direct sum. The modularity presented is of value not only because it enables us to build a complex term rewriting system from its simple parts in automated theorem provers, functional programs, and algebraic specifications, but also because of the analysis of various modular structures appearing in similar systems.

(2) A simple method has been proposed for proving the equivalence of two given term rewriting systems without the explicit use of induction, extending the *inductionless induction* methods developed by Musser, Gougen, Huet and Hullot into a more general framework. Our results have made it clear that the *inductionless induction* concept essentially requires only two limitations: the Church-Rosser property and reachability. It has been demonstrated that the method can be effectively applied to deriving a new term rewriting system from a given one by using equivalence transformation rules. We believe that our method provides a very useful means of proving the equivalence which arises in various formal systems.

(3) To study the Church-Rosser property of non-linear systems, a new type of term rewriting system, i.e., a membership-conditional system, has been introduced. It has been demonstrated that with appropriate membership conditions, non-linear systems behave like left-linear systems. Based on this observation, a criterion for the Church-Rosser property has been indicated for restricted non-linear systems. The result shows that the membership-conditional rewriting technique is useful in constructing Church-Rosser term rewriting systems having non-linear rules.

Appendix A

Proof of Lemma 2.3.2.8

In this appendix, we prove Lemma 2.3.2.8 in Chapter 2: If $root(M\downarrow) \in F_d$ then $|E_d(M)| = 1$. We need to prove many lemmas before achieving our goal.

We write $z \in M$ if the variable occurrence z is in the term M; otherwise $z \notin M$.

A.1 Lemma. Let $z \notin C[P]$ and let $C[P] \xrightarrow[pull]{*} P$. Then $z \in C[z]\downarrow$.

Proof. Let $C[e(P)] \xrightarrow[e']{k} e(P)$. Then we prove the lemma by induction on k. The case $k = 0$ is trivial. Assume the lemma for $k-1$ $(k > 0)$. We will show the lemma for k.

Let $C[e(P)] \xrightarrow[e']{} C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{k-1} e(P)$. From Lemma 2.3.2.2, we can have a reduction $C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{k'} e(P)$ $(k' \leq k-1)$. By induction hypothesis, $z \in C'[P, \cdots, z, \cdots, P]\downarrow$. Since $\rightarrow$ is confluent, $C[z]\downarrow \equiv C'[z, \cdots, z, \cdots, z]\downarrow$. If $z \notin C'[z, \cdots, z, \cdots, z]\downarrow$, then $z \notin C'[P, \cdots, z, \cdots, P]\downarrow$; this provides a contradiction to $z \in C'[P, \cdots, z, \cdots, P]\downarrow$. Therefore $z \in C[z]\downarrow$. $\Box$

If $C[P] \xrightarrow{*} C'[P]$ is a reduction such that $C[e(P)] \xrightarrow[e']{*} C'[e(P)]$, then the occurrence P in $C[P]$ is called an ancestor of the occurrence P in $C'[P]$.

A.2. Lemma. Let $C[P] \xrightarrow{*} Q$ where P contains an ancestor of Q as a subterm occurrence and let $z \notin C[P]$. Then $z \in C[z]\downarrow$.

Proof. Since P contains an ancestor of Q, $P \equiv C'[Q]$ and $C[P] \equiv C[C'[Q]] \xrightarrow[pull]{*} Q$ for some context $C'[\ ]$. If $z \notin C[z]\downarrow$ then $z \notin C[C'[z]]\downarrow$; it is contradictory to Lemma A.1. Thus $z \in C[z]\downarrow$. □

We write $C[P, \cdots, P] \xrightarrow[pull]{*} P$ when some P can be pulled up, that is, $C[P, \cdots, P, e(P), P, \cdots, P] \xrightarrow[e']{*} e(P)$ is obtained by replacing some occurrence P in $C[P, \cdots, P]$ with $e(P)$.

A.3. Lemma. Let $C[P] \xrightarrow[pull]{*} P$ and let $C[z]\downarrow \equiv \tilde{C}[z, \cdots, z]$ where $z \notin C[P]$ and $z \notin \tilde{C}[\ , \cdots, \ ]$. Then $\tilde{C}[P, \cdots, P] \xrightarrow[pull]{*} P$. Note that $z \in \tilde{C}[z, \cdots, z]$ from Lemma A.1.

Proof. Let $C[e(P)] \xrightarrow[e']{k} e(P)$. Then we prove the lemma by induction on k. The case $k = 0$ is trivial. Assume the lemma for $k-1$ $(k > 0)$. We will show the lemma for k.

Let $C[e(P)] \xrightarrow[e']{} C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{k-1} e(P)$. From Lemma 2.3.2.2, we can have a reduction $C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{k'} e(P)$ $(k' \leq k-1)$.
Take $C'[P, \cdots, z, \cdots, P]\downarrow \equiv C''[z, \cdots, z]$ where $z \notin C''[\ , \cdots, \ ]$. Note that $z \in C''[z, \cdots, z]$ from Lemma A.1. By induction hypothesis,
$C''[P, \cdots, P, e(P), P, \cdots, P] \xrightarrow[e']{*} e(P)$.

From $C'[z, \cdots, z, \cdots, z]\downarrow \equiv \tilde{C}[z, \cdots, z]$, $C'[P, \cdots, z, \cdots, P]\downarrow \equiv C''[z, \cdots, z]$, and the confluence property of $\rightarrow$, we can easily show that $\tilde{C}[P, \cdots, P] \xrightarrow{*} C''[P, \cdots, P]$. By tracing this reduction, we can also obtain the reduction $\tilde{C}[e(P), \cdots, e(P)] \xrightarrow[e']{*} C''[e(P), \cdots, e(P)]$.

Thus $\tilde{C}[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{*} e(P)$. From Lemma 2.3.2.2, it follows that $\tilde{C}[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{*} e(P)$. □

A.4. Lemma. Let $C[P] \xrightarrow[pull]{*} P$ and let $C[z] \xrightarrow{*} C'[z, \cdots, z]$ where $z \notin C[P]$ and $z \notin C'[\ , \cdots, \ ]$. Then $C'[P, \cdots, P] \xrightarrow[pull]{*} P$. Note that $z \in C'[z, \cdots, z]$ from Lemma A.1.

Proof. Let $C[z] \downarrow\equiv C'[z, \cdots, z] \downarrow\equiv \tilde{C}[z, \cdots, z]$ where $z \notin \tilde{C}[\ , \cdots, \ ]$. Then $C'[e(P), \cdots, e(P)] \xrightarrow{*} \tilde{C}[e(P), \cdots, e(P)]$. From Lemma A.3, $\tilde{C}[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{*} e(P)$. Thus we can obtain $C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{*} e(P)$. From Lemma 2.3.2.2, it follows that $C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{*} e(P)$. □

A.5. Lemma. Let $C[P] \xrightarrow[pull]{*} P$ and let $P \xrightarrow{*} Q$. Then $C[Q] \xrightarrow[pull]{*} Q$.

Proof. Let $C[e(P)] \xrightarrow[e']{k} e(P)$. Then we prove the lemma by induction on k. The case $k = 0$ is trivial. Assume the lemma for $k-1$ $(k > 0)$. We will show the lemma for k.

Let $C[e(P)] \xrightarrow[e']{} C'[e(P), \cdots, e(P), \cdots, e(P)] \xrightarrow[e']{k-1} e(P)$. From Lemma 2.3.2.2, we can have a reduction $C'[P, \cdots, e(P), \cdots, P] \xrightarrow[e']{k'} e(P)$ $(k' \leq k-1)$. By induction hypothesis, $C'[P, \cdots, e(Q), \cdots, P] \xrightarrow[e']{*} e(Q)$. From Lemma A.4, we have $C'[Q, \cdots, e(Q), \cdots, Q] \xrightarrow[e']{*} e(Q)$. Since $C[e(Q)] \xrightarrow[e']{} C'[e(Q), \cdots, e(Q), \cdots, e(Q)] \xrightarrow[e']{*} C'[Q, \cdots, e(Q), \cdots, Q]$, the lemma holds. □

A.6. Lemma. Let $C[P] \xrightarrow{*} Q$ where P contains an ancestor of Q and let $C[z] \xrightarrow{*} C'[z, \cdots, z]$ where $z \notin C[P]$ and $z \notin C'[\ , \cdots, \ ]$. Then $C'[P, \cdots, P] \xrightarrow{*} Q$ where some occurrence P contains an ancestor of Q. Note that $z \in C'[z, \cdots, z]$ from Lemma A.2.

Proof. Let $P \equiv C''[Q]$. Then $C[C''[z]] \xrightarrow{*} C'[C''[z], \cdots, C''[z]]$. From Lemma A.4, we can show that $C'[C''[Q], \cdots, C''[Q], \cdots, C''[Q]] \xrightarrow[pull]{*} Q$. □

A.7. Lemma. Let $M \equiv C[\![M_1, \cdots, M_m]\!]$ where $M_i \in NF$ for all i and let

$M' \equiv C[z_1, \cdots, z_m]$ where $z_1, \cdots, z_m$ are fresh variables not in M. If M has an infinite reduction $M \to\to\to \cdots$, then M' has an infinite reduction $M' \to\to\to \cdots$.

Proof. It is trivial from the definition of the direct sum. □

A.8. Lemma. Let $M \equiv C[\![M_1, \cdots, M_p, \cdots, M_m]\!] \xrightarrow{*} Q$ where $root(M) \in F_{\bar{d}}$, $root(M\ \downarrow), root(Q) \in F_d$. Let $M_p \notin E_d(M)$ and M_p contains an ancestor of Q. Then $root(M_p\ \downarrow) \in F_{\bar{d}}$ and there exists some $P \in E_{\bar{d}}(M_p)$ such that $M' \equiv C[M_1, \cdots, P, \cdots, M_m] \xrightarrow{*} Q$ where P contains an ancestor of Q.

Proof. Let $A_0 \equiv C[M_1, \cdots, M_{p-1}, z, M_{p+1}, \cdots, M_m]\downarrow \quad (z \notin M)$. Set $i := 0$ and consider the following algorithm:

(1) If A_i contains precisely one occurrence z then the algorithm terminates with output A_i.

(2) Let $A_i \equiv C_i[z, \cdots, z, \cdots, z]\ (z \notin C_i)$. From Lemma A.6, $C_i[M_p, \cdots, M_p, \cdots, M_p] \xrightarrow{*} Q$ where some occurrence M_p contains an ancestor of Q. By replacing the occurrence M_p containing an ancestor of Q with z, we obtain $C_i[M_p, \cdots, z, \cdots, M_p]$. Let $A_{i+1} \equiv C_i[M_p, \cdots, z, \cdots, M_p]\downarrow$. Note that $z \in A_{i+1}$ from Lemma A.2.

(3) Set $i := i + 1$. Go to (1).

We will show that the algorithm terminates for any A_0. Let the substitution $\theta = [z := M_p\downarrow]$. From the confluence property of $\to$, $A_{i+1} \equiv C_i[M_p, \cdots, z, \cdots, M_p]\downarrow \equiv C_i[M_p\ \downarrow, \cdots, z, \cdots, M_p\ \downarrow]\ \downarrow$. If the algorithm produces an infinite sequence $A_0, A_1, A_2, \cdots$, then we can obtain an infinite reduction $A_0\theta \xrightarrow{+} A_1\theta \xrightarrow{+} A_2\theta \xrightarrow{+} \cdots$. Note that we can write $A_0\theta \equiv \tilde{C}[\![N_1, \cdots, N_n]\!]$ where $N_i \in NF$ for all i. Thus, from Lemma A.7, we have an infinite reduction $\tilde{C}[z_1, \cdots, z_n] \to\to\to \cdots$: a contradiction to termination of $R_{\bar{d}}$. Therefore the algorithm must terminate eventually.

Now let $A_N \equiv C_N[z]$ be an output of the algorithm. Since $M_p \notin E_d(M)$, $A_N \not\equiv z$. Note that $C_N[z] \in NF$ and $C_N[M_p] \xrightarrow{*} Q$ where the occurrence M_p contains an ancestor of Q. We can write $C_N[M_p] \equiv C'[\![P_1, \cdots, P_k, M_p, P_{k+1}, \cdots, P_r]\!]$ where $C_N[z] \equiv C'[P_1, \cdots, P_k, z, P_{k+1}, \cdots, P_r]$.

Suppose $root(M_p \downarrow) \notin F_{\bar{d}}$. Then $M \downarrow \equiv C_N[M_p \downarrow]$; it follows that $root(M \downarrow) \in F_{\bar{d}}$. It is contradictory to $root(M \downarrow) \in F_d$. Hence $root(M_p \downarrow) \in F_{\bar{d}}$.

Since $C_N[z] \in NF$, $root(C_N[M_p]) \in F_{\bar{d}}$, $root(Q) \in F_d$, and $C_N[M_p] \equiv C'[\![P_1, \cdots, P_k, M_p, P_{k+1}, \cdots, P_r]\!] \xrightarrow{*} Q$; there exists some P' such that $M_p \xrightarrow{*} P'$, $root(P') \in F_{\bar{d}}$, and
$C'[\![P_1, \cdots, P_k, M_p, P_{k+1}, \cdots, P_r]\!] \xrightarrow{*} C'[P_1, \cdots, P_k, P', P_{k+1}, \cdots, P_r] \xrightarrow{*} Q$
where P' contains an ancestor of Q. From Lemma 2.3.2.6, we have $P \in E_{\bar{d}}(M_p)$ such that $P \xrightarrow{*} P'$. Thus it follows that $C_N[P] \xrightarrow{*} Q$ where P also contains an ancestor of Q.

Since $\rightarrow$ is confluent, $P \downarrow \equiv M_p \downarrow$; hence, it can be obtained that $M' \equiv C[M_1, \cdots, P, \cdots, M_m] \xrightarrow{*} C_N[P]$ where the occurrence P in M' is an ancestor of the occurrence P in $C_N[P]$. Therefore the lemma holds. □

A.9. Lemma. Let $C[\![M_1, \cdots, M_p, \cdots, M_m]\!] \xrightarrow[pull]{*} M_p$ where $M_i \in NF$ for all i. Then $C[z_1, \cdots, z_p, \cdots, z_m] \downarrow \equiv z_p$.

Proof. Obvious from the definition of the direct sum. □

$N \subset M$ stands for N is a subterm of M and $N \not\equiv M$.

A.10. Lemma. Let $root(M) \in F_{\bar{d}}$, $root(M \downarrow) \in F_d$, $M \equiv C[\![M_1, \cdots, M_p, \cdots, M_m]\!]$. Let $M_p \in E_d(M)$. Then $\forall Q \subset M_p,\ Q \notin E_d(M)$.

Proof. Since $M_p \downarrow \equiv M \downarrow$, $root(M_p \downarrow) \in F_d$. Thus we can write

$C[M_1\downarrow, \cdots, M_p\downarrow, \cdots, M_m\downarrow] \equiv C'[\![N_1, \cdots, N_{k-1}, M_p\downarrow, N_{k+1}, \cdots, N_n]\!]$

where $N_i \in NF$ for any i. From Lemmas A.4 and A.5,

$C'[\![N_1, \cdots, N_{k-1}, M_p\downarrow, N_{k+1}, \cdots, N_n]\!] \xrightarrow[pull]{*} M_p\downarrow$.

Applying Lemma A.9, $C'[z_1, \cdots, z_{k-1}, z_k, z_{k+1}, \cdots, z_n]\downarrow \equiv z_k$. Thus

$C[M_1, \cdots, M_{p-1}, z_k, M_{p+1}, \cdots, M_m] \xrightarrow{*} C'[N_1, \cdots, N_{k-1}, z_k, N_{k+1}, \cdots, N_n]\downarrow \equiv z_k$.

Suppose an occurrence Q ($Q \subset M_p$) is in $E_d(M)$. Then, from Lemma A.6, $M_p \xrightarrow{+} Q$. This is contradictory to $Q \in E_d(M)$. Hence the lemma holds. □

A.11. Lemma. Let $root(M) \in F_{\bar{d}}$, $root(M\downarrow) \in F_d$, $M \equiv C[\![M_1, \cdots, M_p, \cdots, M_m]\!]$. Let $M_p \in E_d(M)$. Then $M_i \notin E_d(M)$ $(i \neq p)$.

Proof. Assume $M_q \in E_d(M)$ for some q $(q \neq p)$. Since $M_p\downarrow \equiv M_q\downarrow \equiv M\downarrow$, it follows that $root(M_p\downarrow), root(M_q\downarrow) \in F_d$. Thus we can write

$C[M_1\downarrow, \cdots, M_p\downarrow, \cdots, M_q\downarrow, \cdots, M_m\downarrow] \equiv$

$C'[\![N_1, \cdots, N_{k-1}, M_p\downarrow, N_{k+1}, \cdots, N_{s-1}, M_q\downarrow, N_{s+1}, \cdots, N_n]\!]$

where $N_i \in NF$ for any i. Using the same discussion as in Lemma A.10, it can be shown that $C'[z_1, \cdots, z_{k-1}, z_k, z_{k+1}, \cdots, z_{s-1}, z_s, z_{s+1}, \cdots, z_n]\downarrow \equiv z_k \equiv z_s$ where $z_k \not\equiv z_s$: a contradiction to the confluence property of $\rightarrow$. Thus it follows that $M_i \notin E_d(M)$ $(i \neq p)$. □

A.12. Theorem (*Lemma 2.3.2.8*). If $root(M\downarrow) \in F_d$ then $|E_d(M)| = 1$.

Proof. The number of the special subterm occurrences in M is inductively defined as follows:

$$\|M\| = \begin{cases} 1 & \text{if } M \in T(F_d, V) \text{ for some } d, \\ 1 + \Sigma_i \|M_i\| & \text{if } M \equiv C[\![M_1, \cdots, M_m]\!] \ \ (m > 0). \end{cases}$$

We will prove the theorem by induction on $\|M\|$. The case $\|M\| = 1$ is trivial. Assume the theorem for $\|M\| < k$ $(k > 1)$, then we will show the case $\|M\| = k$. If $root(M) \in F_d$ then the above property is trivial since $E_d(M) = \{M\}$. Thus we

consider only the non trivial case of $root(M) \in F_{\bar{d}}$. Let $M \equiv C[\![M_1, \cdots M_p, \cdots, M_m]\!]$ $(m > 0)$.

Case 1. $\forall Q \in E_d(M), \exists M_i$, an ancestor of Q is M_i.

The theorem holds from Lemma A.11.

Case 2. $\exists Q \in E_d(M), \forall M_i$, no ancestor of Q is M_i.

Then there exists some M_p such that M_p contains an ancestor of Q and $M_p \not\equiv Q$. Note that $M_p \notin E_d(M)$ from Lemma A.10. From Lemma A.8, $root(M_p \downarrow) \in F_{\bar{d}}$; by induction hypothesis, $|E_{\bar{d}}(M_p)| = 1$. Say $E_{\bar{d}}(M_p) = \{P\}$. Let $M' \equiv C[M_1, \cdots, P, \cdots, M_m]$. Then, from Lemma A.8 and the uniqueness of P, we can show that $Q \in E_d(M')$ and P must contain an ancestor of Q. Furthermore, from the uniqueness of P, it follows that for any $Q' \in E_d(M)$ if M_p contains an ancestor of Q' then $Q' \in E_d(M')$ and P contains an ancestor of Q'.

Any element in $E_d(M)$ such that M_p does not contain an ancestor of the element is in $E_d(M')$ from Lemma A.4. Thus $E_d(M) = E_d(M')$.

From $\|M'\| < \|M\|$ and induction hypothesis, it follows that $|E_d(M)| = |E_d(M')| = 1$. □

Appendix B

Proof of Reachability

In this appendix, we present a simple method for verifying reachability in Chapter 4 by using the idea of the test set [14, 38, 45, 50, 51, 77, 89]. From here on we assume that term rewriting system R is left-linear.

We first explain the notations. Let $C[\ ,\ldots,\ ]$ be a context and let $T_1,\cdots,T_n$ $(n \geq 0)$ be the non-empty sets of terms. Then the set of terms $C[T_1,\ldots,T_n] = \{C[M_1,\ldots M_n] \mid\ M_i \in T_i\}$. A substitution φ is a mapping from a term set T to a power set 2^T such that for a term $M \in T$, $\varphi(M) \subseteq T$ is completely determined by its values $\varphi(x),\ldots,\varphi(z) \subseteq T$ on the variable symbols $x,\ldots,z$ occurring in M. We define $\varphi(M) = \{M\}$ if no variable symbol occurs in M. $\varphi(C)[T_1,\ldots,T_n]$ denotes the term set obtained by replacing the variable symbols $x,\ldots,z$ and the holes $\Box$ occurring in $C[\ ,\ldots,\ ]$ with $\varphi(x),\ldots,\varphi(z)$ and $T_1,\ldots,T_n$, respectively. A finite set of linear terms $S = \{M_1,\ldots,M_s\}$ is a test set for a term set T with φ if $T = \bigcup_i \varphi(M_i)$. For example, $S = \{x+0, x+s(y)\}$ is a test set for the term set $\mathbf{N}+\mathbf{N}$ with $\varphi(x) = \varphi(y) = \mathbf{N}$. Here, $\mathbf{N} = T(\{0,s\})$. For more discussions concerning the test set, see [14, 38, 45, 50, 51, 77, 89].

We say a well-founded partial ordering $>$ on a term set T is stable under substitution if $M\theta > N\theta$ for any θ and each $M > N$. Here, θ is a substitution from T to T, and $M, N \in T$. For example, let l be a fixed positive number and let $\underset{l}{>}$ be the

well-founded ordering on a term set T defined by $f(M_1, \ldots, M_n) \underset{l}{>} f(N_1, \ldots, N_n)$ iff $N_l \subseteq M_l$ and $N_l \not\equiv M_l$ (i.e., N_l is a proper subterm of M_l). Then, it is obvious that $\underset{l}{>}$ is stable under substitution.

We often write $\vec{T}$ for a direct product of sets $T \times \cdots \times T$ and $\vec{M}$ for an ordered n-tuples $\langle M_1, \ldots, M_n \rangle \in T_1 \times \cdots \times T_n$. We write $f(\vec{M})$ for $f(M_1, \ldots, M_n)$ and $f(\vec{T})$ for $f(T, \ldots, T)$.

$T \Rightarrow T'$ denotes that T' is reachable from T under $\xrightarrow{*}$ (and also $=$).

The following properties can be easily demonstrated from the above definitions.

B.1. Properties.

(1) If $T \subseteq T'$ then $T \Rightarrow T'$.

(2) If $T \Rightarrow T'$ and $T' \Rightarrow T''$ then $T \Rightarrow T''$.

(3) If $T_i \Rightarrow T_i'$ $(i = 1, \ldots, n)$ then $C[T_1, \ldots, T_n] \Rightarrow C[T_1', \ldots, T_n']$.

(4) If $f(\overrightarrow{T(F)}) \Rightarrow T(F)$ then $T(F \cup \{f\}) \Rightarrow T(F)$.

(5) If $T(F \cup \{f\}) \Rightarrow T$ and $T(F \cup \{f'\}) \Rightarrow T$ then $T(F \cup \{f, f'\}) \Rightarrow T$.

(6) If $M \xrightarrow{*} N$ then $\varphi(M) \Rightarrow \varphi(N)$.

We can prove reachability for the examples in Chapter 4 by using the properties and the following theorem. Note that our method does not assume the strongly normalizing property for term rewriting systems.

B.2. Theorem. Let R be a left-linear term rewriting system and let $>$ be a well-founded partial ordering on a term set that is stable under substitution. Let $S = \{f(\vec{P_1}), \ldots, f(\vec{P_p})\}$ be a test set for $f(\overrightarrow{T(F)})$ with $\varphi(x) = T(F)$ for all $x \in V$.

Then $f(\overrightarrow{T(F)}) \Rightarrow T$ if R satisfies the following conditions: For any $f(\vec{P_i}) \in S$ there exists a term Q_i such that

$f(\vec{P_i}) \xrightarrow{*} Q_i \equiv C[f(\vec{A_1}), \ldots, f(\vec{A_q})]$ $(q \geq 0)$

where C contains no function symbol f and

(1) $f(\vec{P_i}) > f(\vec{A_j})$ for all j,

(2) $f(\vec{A_j}) \in f(T(\overrightarrow{F, V}))$ for all j,

(3) $\varphi(C)[T, \ldots, T] \Rightarrow T$.

Proof. Make the rewriting rules $f(\vec{P_i}) \triangleright Q_i$ from each pair of $f(\vec{P_i})$ and Q_i $(i = 1, \ldots, p)$. Let R_S be the term rewriting system defined by these rules. Then, $\underset{S}{\rightarrow} \subseteq \xrightarrow{*}$. Note that any $f(\vec{M}) \in f(\overrightarrow{T(F)})$ is a $\underset{S}{\rightarrow}$ redex since S is a test set for $f(\overrightarrow{T(F)})$.

Let $M_0 \in f(\overrightarrow{T(F)})$. First, we show that there is no infinite reduction sequence $M_0 \underset{S}{\rightarrow} M_1 \underset{S}{\rightarrow} M_2 \underset{S}{\rightarrow} \cdots$. Let $OC(M_i)$ denote the multiset of the $\underset{S}{\rightarrow}$ redex occurrences in M_i. From the stability under substitution of $>$ and condition (1) in the theorem, $OC(M_i) \gg OC(M_{i+1})$ is obtained. Here, $\gg$ is the multiset ordering extended from $>$. $\gg$ is well-founded since $>$ is so [17]. Thus, the above reduction sequence must terminate eventually into a $\underset{S}{\rightarrow}$ normal form. Note that the $\underset{S}{\rightarrow}$ normal form contains no term in $f(\overrightarrow{T(F)})$ as a subterm occurrence. We write $d(M_0)$ for the maximal length of the $\underset{S}{\rightarrow}$ reduction sequences starting with M_0 into a $\underset{S}{\rightarrow}$ normal form of M_0.

Now, we show $\forall f(\vec{M}) \in f(\overrightarrow{T(F)}), \exists N \in T, f(\vec{M}) \xrightarrow{*} N$ by induction on $d(f(\vec{M}))$.

Basis ($d(f(\vec{M})) = 1$). There exist some $f(\vec{P_i}) \triangleright Q_i \in R_S$ and some θ such that $f(\vec{M}) \equiv f(\vec{P_i})\theta \underset{S}{\rightarrow} Q_i\theta$, where $Q_i\theta$ is a $\underset{S}{\rightarrow}$ normal form. Since $Q_i\theta$ contains no term in $f(\overrightarrow{T(F)})$, $\varphi(Q_i) \Rightarrow T$ follows from condition (3) in the theorem. From $f(\vec{P_i})\theta \in f(\overrightarrow{T(F)})$ and $f(\vec{P_i}) \xrightarrow{*} Q_i$, we can show $Q_i\theta \in \varphi(Q_i)$. Therefore, there exists $N \in T$ such that $f(\vec{M}) \xrightarrow{*} Q_i\theta \xrightarrow{*} N$.

Induction. Assume that the theorem is true for $d(f(\vec{M})) \leq k$ $(k \geq 1)$. Now, we will show the case for $d(f(\vec{M})) = k+1$. There exist some $f(\vec{P_i}) \triangleright Q_i \in R_S$ and some θ such that $f(\vec{M}) \equiv f(\vec{P_i})\theta \underset{S}{\rightarrow} Q_i\theta \equiv C[f(\vec{A_1}), \ldots, f(\vec{A_q})]\theta \equiv C\theta[f(\vec{A_1})\theta, \ldots, f(\vec{A_q})\theta]$ $(q \geq 1)$. From condition (2) in the theorem, $f(\vec{A_j})\theta \in f(\overrightarrow{T(F)})$ for all j. From the inductive hypothesis, we can obtain $N_j \in T$ $(j = 1, \ldots, q)$ such that $f(\vec{A_j})\theta \xrightarrow{*} N_j$ since $d(f(\vec{M})) > d(f(\vec{A_j})\theta)$.

Thus, it follows that $f(\vec{M}) \xrightarrow{*} C\theta[f(\vec{A_1})\theta, \ldots, f(\vec{A_q})\theta] \xrightarrow{*} C\theta[N_1, \ldots, N_q] \in \varphi(C)[T, \ldots, T]$. Therefore, by condition (3) in the theorem, there exists $N \in T$ such that
$C\theta[N_1, \ldots, N_q] \xrightarrow{*} N$. □

We now illustrate how to prove reachability by using the above theorem.

B.3. Example. Consider the term rewriting system R_1 in Example 4.3.1 to be R.

$$R \quad \begin{cases} x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

We will prove $T(\{+, s, 0\}) \Rightarrow \mathbf{N}$. From Property B.1(4), it is sufficient to prove $\mathbf{N} + \mathbf{N} \Rightarrow \mathbf{N}$ by using the theorem. Take the test set $S = \{x + 0, x + s(y)\}$ for $\mathbf{N} + \mathbf{N}$ with $\varphi(x) = \varphi(y) = \mathbf{N}$ and the well-founded ordering $\underset{2}{>}$. Note that $\underset{2}{>}$ is stable under substitution. We must verify the conditions in the theorem for $x + 0$ and $x + s(y)$ respectively.

For $x + 0$, $x + 0 \xrightarrow{*} x$. Consider the pair of $x + 0$ and x. Then, the pair trivially satisfies conditions (1) and (2) in the theorem. Condition (3) is also satisfied since $\varphi(x) = \mathbf{N}$.

For $x + s(y)$, $x + s(y) \xrightarrow{*} s(x + y)$. We must verify the conditions in the theorem

for the pair of $x + s(y)$ and $s(x + y) \equiv s(\Box)[x + y]$. Condition (1) is satisfied since $x + s(y) \underset{2}{>} x + y$. Condition (2) is satisfied since $x + y \in T(\{s, 0\}, V) + T(\{s, 0\}, V)$. Since $\varphi(s(\Box))[\mathbf{N}] = s(\mathbf{N})$, condition (3) is satisfied. Therefore, $\mathbf{N} + \mathbf{N} \Rightarrow \mathbf{N}$ follows from the theorem. $\Box$

B.4. Example. Consider the term rewriting system R_1 in Example 4.4.4 to be R. Note that R does not have the strongly normalizing property.

$$R \quad \begin{cases} d(x) \triangleright \mathit{if}(x, 0, s(s(d(x - s(0))))) \\ h(x) \triangleright \mathit{if}(x, 0, \mathit{if}(x - s(0), 0, s(h(x - s(s(0)))))) \\ \mathit{if}(0, y, z) \triangleright y \\ \mathit{if}(s(x), y, z) \triangleright z \\ x - 0 \triangleright x \\ s(x) - s(y) \triangleright x - y \end{cases}$$

We will prove $T(\{d, h, s, 0\}) \Rightarrow \mathbf{N}$. From Property B.1(5), it is sufficient to prove $T(\{d, s, 0\}) \Rightarrow \mathbf{N}$ and $T(\{h, s, 0\}) \Rightarrow \mathbf{N}$.

Let us prove $T(\{d, s, 0\}) \Rightarrow \mathbf{N}$. From Property B.1(4), it is sufficient to prove $d(\mathbf{N}) \Rightarrow \mathbf{N}$ by using the theorem. Take the test set $S = \{d(0), d(s(x))\}$ for $d(\mathbf{N})$ with $\varphi(x) = \mathbf{N}$ and the substitution preserved well-founded ordering $\underset{1}{>}$. We must verify the conditions in the theorem for $d(0)$ and $d(s(x))$, respectively.

For $d(0)$, $d(0) \xrightarrow{*} 0$. Consider the pair of $d(0)$ and 0. Then, the conditions are trivially satisfied.

For $d(s(x))$, $d(s(x)) \xrightarrow{*} s(s(d(x)))$. Consider the pair of $d(s(x))$ and $s(s(d(x))) \equiv s(s(\Box))[d(x)]$. Condition (1) is satisfied since $d(s(x)) \underset{1}{>} d(x)$. Condition (2) is satisfied since $d(x) \in d(T(\{s, 0\}, V))$. Since $\varphi(s(s(\Box)))[\mathbf{N}] = s(s(\mathbf{N}))$, condition (3) is also satisfied. Hence, $T(\{d, s, 0\}) \Rightarrow \mathbf{N}$ follows from the theorem.

Next, let us show $T(\{h, s, 0\}) \Rightarrow \mathbf{N}$. From property B.1(4), it is sufficient to prove

$h(\mathbf{N}) \Rightarrow \mathbf{N}$ by using the theorem. Take the test set $S = \{h(0), h(s(0)), h(s(s(x)))\}$ for $h(\mathbf{N})$ with $\varphi(x) = \mathbf{N}$ and the well-founded ordering $\underset{1}{>}$. Note that $\underset{1}{\geq}$ is stable under substitution. We need to verify the conditions in the theorem for $h(0), h(s(0))$, and $h(s(s(x)))$, respectively. For $h(0)$ and $h(s(0))$, it can be easily shown that the conditions are satisfied.

For $h(s(s(x)))$, $h(s(s(x))) \xrightarrow{*} s(h(x))$. Consider the pair of $h(s(s(x)))$ and $s(h(x)) \equiv s(\Box)[h(x)]$. Condition (1) is satisfied since $h(s(s(x))) \underset{1}{>} h(x)$. Condition (2) is satisfied since $h(x) \in h(T(\{s, 0\}, V))$. Since $\varphi(s(\Box))[\mathbf{N}] = s(\mathbf{N})$, condition (3) is satisfied. Hence, $T(\{h, s, 0\}) \Rightarrow \mathbf{N}$ follows from the theorem. $\Box$

B.5. Example. Consider the term rewriting system R_2 in Example 4.5.5 to be R.

$$R \quad \begin{cases} f(0) \triangleright s(0) \\ f(s(0)) \triangleright s(0) \\ f(s(s(x))) \triangleright p(h(g(x))) \\ g(0) \triangleright \langle s(0), s(0) \rangle \\ g(s(x)) \triangleright h(g(x)) \\ h(x) \triangleright \langle p(x) + q(x), p(x) \rangle \\ p(\langle x, y \rangle) \triangleright x \\ q(\langle x, y \rangle) \triangleright y \\ x + 0 \triangleright x \\ x + s(y) \triangleright s(x + y) \end{cases}$$

We will prove $T(\{f, +, s, 0\}) \Rightarrow \mathbf{N}$. From $T(\{+, s, 0\}) \Rightarrow \mathbf{N}$ and Property B.1(5), it is sufficient to prove $T(\{f, s, 0\}) \Rightarrow \mathbf{N}$. From Property B.1(4), it is sufficient to prove $f(\mathbf{N}) \Rightarrow \mathbf{N}$. Take the test set $S = \{f(0), f(s(0)), f(s(s(x)))\}$ for $f(\mathbf{N})$ with $\varphi(x) = \mathbf{N}$ and the well-founded ordering $\underset{1}{>}$. Note that $\underset{1}{\geq}$ is stable under substitution.

We must verify the conditions in the theorem for $f(0)$, $f(s(0))$ and $f(s(s(x)))$, respectively. For $f(0), f(s(0))$, it is obvious that the conditions are satisfied.

For $f(s(s(x)))$, $f(s(s(x))) \xrightarrow{*} p(h(g(x)))$. Consider the pair of $f(s(s(x)))$ and $p(h(g(x)))$. Then, conditions (1) and (2) are trivially satisfied. For condition (3), we must prove $\varphi(p(h(g(x)))) = p(h(g(\mathbf{N}))) \Rightarrow \mathbf{N}$. If $g(\mathbf{N}) \Rightarrow \langle \mathbf{N}, \mathbf{N} \rangle$ then the condition is satisfied since $p(h(g(\mathbf{N}))) \Rightarrow p(h(\langle \mathbf{N}, \mathbf{N} \rangle)) \Rightarrow p(\langle p(\langle \mathbf{N}, \mathbf{N} \rangle) + q(\langle \mathbf{N}, \mathbf{N} \rangle), p(\langle \mathbf{N}, \mathbf{N} \rangle) \rangle)$ $\Rightarrow p(\langle \mathbf{N} + \mathbf{N}, \mathbf{N} \rangle) \Rightarrow \mathbf{N} + \mathbf{N} \Rightarrow \mathbf{N}$.

Now, we prove $g(\mathbf{N}) \Rightarrow \langle \mathbf{N}, \mathbf{N} \rangle$ by using the theorem. Take the test set $S = \{g(0), g(s(x))\}$ for $g(\mathbf{N})$ with $\varphi(x) = \mathbf{N}$ and the well-founded ordering $\underset{1}{>}$. Note that $\underset{1}{>}$ is stable under substitution. We must verify the conditions in the theorem for $g(0)$ and $g(s(x))$, respectively.

For $g(0)$, $g(0) \xrightarrow{*} \langle s(0), s(0) \rangle$. Consider the pair of $g(0)$ and $\langle s(0), s(0) \rangle$. Then, conditions (1) and (2) are trivially satisfied. Condition (3) follows from $\varphi(\langle s(0), s(0) \rangle)$ $= \{\langle s(0), s(0) \rangle\} \Rightarrow \langle \mathbf{N}, \mathbf{N} \rangle$.

For $g(s(x))$, $g(s(x)) \xrightarrow{*} h(g(x))$. Consider the pair of $g(s(x))$ and $h(g(x)) \equiv h(\Box)[g(x)]$. Then, condition (1) is satisfied since $g(s(x)) \underset{1}{>} g(x)$. Condition (2) is satisfied since $g(x) \in g(T(\{s, 0\}, V))$. Condition (3) follows from $\varphi(h(\Box))[\langle \mathbf{N}, \mathbf{N} \rangle] =$ $h(\langle \mathbf{N}, \mathbf{N} \rangle) \Rightarrow \langle p(\langle \mathbf{N}, \mathbf{N} \rangle) + q(\langle \mathbf{N}, \mathbf{N} \rangle), p(\langle \mathbf{N}, \mathbf{N} \rangle) \rangle \Rightarrow \langle \mathbf{N} + \mathbf{N}, \mathbf{N} \rangle \Rightarrow \langle \mathbf{N}, \mathbf{N} \rangle$. $\Box$

Appendix C
Fast Knuth-Bendix Completion with a Term Rewriting System Compiler

A term rewriting system compiler can greatly improve the execution speed of reductions by transforming rewriting rules into target code. In this appendix, we present a new application of the term rewriting system compiler: the Knuth-Bendix completion algorithm. The compiling technique proposed in this algorithm, is dynamic in the sense that rewriting rules are repeatedly compiled in the completion process. The execution time of the completion with dynamic compiling is ten or more times as fast as that obtained with a traditional term rewriting system interpreter [102].

C.1. Introduction

A term rewriting system compiler translates rewriting rules into target code in another language such as LISP [47, 91], PASCAL [29], C [84], or an assembly language [42]. The execution time of compiled code is usually hundreds or thousands of times as fast as that of the corresponding term rewriting system interpreter.

In this appendix, we propose a new application of a term rewriting system com-

piler: the Knuth-Bendix completion algorithm [59]. The Knuth-Bendix completion algorithm is well known as a useful technique to solve the word problem of an equational theory [22, 27, 39, 59]. However, as far as the author knows, the Knuth-Bendix algorithm has up to now only been executed with a term rewriting system interpreter. The reason why it has never been executed with a term rewriting system compiler is as follows:

(1) In the completion process, rewriting rules are repeatedly modified; hence, they must be recompiled each time. This dynamic compiling is difficult for most term rewriting system compilers which cannot produce compiled code quickly.

(2) Most term rewriting system compilers have been developed for functional programming languages or algebraic specifications of which rules are restricted by some properties, such as left-linearity, non-ambiguity (i.e. the non-overlapping property), and strong sequentiality [29, 42, 73, 84]. On the other hand, the completion process must treat any rules without such restrictions.

Recently, Tomura [91] and Kaplan [47] proposed independently an interesting term rewriting system compiler based on some tricky use of LISP features which translates rewriting rules into LISP functions; for example, the term rewriting system

$$R_{minus} \quad \begin{cases} minus(x, 0) \triangleright x \\ minus(s(x), s(y)) \triangleright minus(x, y) \\ minus(x, x) \triangleright 0 \end{cases}$$

is translated into the LISP function $\underline{minus}$ as follows:

```
(defun minus (X Y)
        (cond [(eq Y 0)     X]
              [(and (eq (car X) 's) (eq (car Y) 's))     (minus (cdr X) (cdr Y))]
              [(equal X Y)     0]
              [t     (list 'minus X Y)] )).
```

Their compiler has advantages for the Knuth-Bendix completion algorithm; it produces compiled code quickly and accepts a wide class of rewriting rules. Furthermore, since the completion process treats only terminating rules, we may use the innermost reduction strategy for computing normal forms which simplifies the compiler for our application. Thus, we can easily apply this compiler to the completion algorithm. Several benchmarks show that the execution time of the completion with dynamic compiling is ten or more times as fast as that obtained with a term rewriting system interpreter [102].

C.2. Benchmarks for TRS Compiler

The purpose of this benchmarks is to compare the performance of a traditional TRS (Term Rewriting System) interpreter and that of the TRS compiler proposed by Kaplan before using them in the Knuth-Bendix completion algorithm. Both systems compute normal forms in the same way, i.e. by the left-innermost reduction strategy [58, 72].

The original TRS compiler proposed by Tomura [91] and Kaplan [47] translates rewriting rules into compiled code through two phases; in the first phase, the TRS compiler generates a LISP function $\underline{f}$ for each function symbol f, and in the second phase, this function $\underline{f}$ is compiled into native machine code by a LISP compiler. However, the execution time of the second phase is usually too long for repeated compiling in the Knuth-Bendix completion process. Thus, our TRS compiler does

not have the second phase; the LISP function $\underline{f}$ generated by the first phase is directly computed by a LISP interpreter.

By using the following rules, the factorial function $fact(n) = n!$ is computed by both systems. Here, natural numbers n are represented in the usual way: $0, s(0), s(s(0)), \cdots$.

$$R_{fact} \begin{cases} plus(x, 0) \triangleright x \\ plus(x, s(y)) \triangleright s(plus(x, y)) \\ times(x, 0) \triangleright 0 \\ times(x, s(y)) \triangleright plus(times(x, y), x) \\ fact(0) \triangleright s(0) \\ fact(s(x)) \triangleright times(s(x), fact(x)) \end{cases}$$

The benchmarks have been made on a TOSHIBA J-3100GT (IBM PC compatible laptop computer with CPU 80286 (8 Mhz)). Both the TRS interpreter and the TRS compiler are written in MuLISP-87. The TRS interpreter is compiled by MuLISP-87 compiler. As stated above, the functions generated by the TRS compiler are not compiled by LISP compiler; they are directly evaluated by MuLISP-87 interpreter. The results are shown in the following table:

	n=0	n=1	n=2	n=3	n=4	n=5	n=6	n=7
T_I (sec.)	0.16	0.55	1.20	3.24	19.67	340.30	∞	∞
T_C (sec.)	0.00	0.00	0.01	0.02	0.07	0.22	1.10	7.30
T_I/T_C	—	—	120	162	281	1546.82	∞	∞

Here, T_I and T_C show the execution time by the interpreter and by the compiled code respectively. ∞ in T_I shows that the computation is impossible because of memory overflow.

C.3. Fast Knuth-Bendix Completion

C.3.1 Completion with Dynamic Compiling

A complete (i.e., confluent and terminating) term rewriting system R which produces the same equality as the one generated by an equational theory E is very important to solve the word problem of E [22, 39, 59]. Knuth and Bendix [59] proposed a famous procedure to find a complete term rewriting system R from a given an equational theory E. According to Dershowitz [19] and Klop [58], a simple version of this procedure is given as follows:

Knuth-Bendix completion algorithm

The procedure has as input a finite set R of rules, a finite set E of equations, and a program to compute a well-founded monotonic ordering $>$ on terms. The critical pairs $\langle P, Q \rangle$ are presented in E as equations $P = Q$.

Repeat while E is not empty. If E is empty, we have successful termination.

(1) Remove an equation $M = N$ (or $N = M$) from E such that $M > N$. If such an equation does not exist, terminate with failure.

(2) Move each rule from R to E whose left-hand side contains an instance of M.

(3) Extend E with all critical pairs in R caused by the rule $M \triangleright N$.

(4) Add $M \triangleright N$ to R.

(5) Use R to normalize the right-hand sides of rules in R.

(6) Use R to normalize both sides of equations in E. Remove each equation that becomes an identical equation.

The Knuth-Bendix completion algorithm spends the greater part of the execution time on (i) computing normal forms and (ii) generating critical pairs. In particular, the completion procedure with a TRS interpreter spends most time on normalizing. Hence, the execution time of the completion can be extremely improved by replacing the TRS interpreter with the TRS compiler described in Section C.1. The procedure with the TRS compiler is given as follows:

Knuth-Bendix completion algorithm with TRS compiler

Repeat while E is not empty. If E is empty, we have successful termination.

(1) Remove an equation $M = N$ (or $N = M$) from E such that $M > N$. If such an equation does not exist, terminate with failure.

(2) Move each rule from R to E whose left-hand side contains an instance of M.

(3) Extend E with all critical pairs in R caused by the rule $M \triangleright N$.

(4) Add $M \triangleright N$ to R.

(5) Compile R into compiled code C.

(6) Use C to normalize the right-hand sides of rules in R.

(7) Use C to normalize both sides of equations in E. Remove each equation that becomes an identical equation.

In the above algorithm, normal forms are computed with compiled code C in (6), (7). (On the other hand, a traditional algorithm computes normal forms with rewriting rules R in interpreter mode. See (5), (6) in the previous algorithm.) Thus, rewriting rules R are repeatedly compiled into compiled code C at (5) while the iteration continues. We call this *dynamic compiling*.

Remark. Purdom and Brown [78] also proposed a *dynamic updating* technique for the Knuth-Bendix completion algorithm different from our *dynamic compiling*. They showed that by repeatedly updating their pattern representation, the number of matching routine called in the completion process can be reduced into about 1/5.

C.3.2. Benchmarks for Completion with TRS Compiler

We compare the execution time of the Knuth-Bendix completion with dynamic compiling, with that of a traditional completion. The examples for the benchmarks are given in the appendix. The benchmarks have been made under the same conditions described in Section C.2. Both the completion algorithms are written in MuLISP-87 and compiled by MuLISP-87 compiler. The results are shown in the following table:

	group	group'	group"	l-r-s	c-group	rev
T_I (sec.)	107.73	130.19	1440.30	222.40	5.27	10.50
T_C (sec.)	10.11	12.14	60.26	15.32	1.43	2.41
T_I/T_C	10.66	10.72	23.90	14.52	3.69	4.36
N	16	18	40	16	3	7

Here, T_I and T_C show the execution time by the completion algorithm with an interpreter and with dynamic compiling, respectively. N shows the number of the compiling times in the completion process. The benchmarks show that the execution time of the completion with dynamic compiling is ten or more times as fast as that with a traditional term rewriting system interpreter.

Appendix

We present the equational theories E [39, 59] used in the benchmarks of Section C.3.2 and the complete term rewriting system R generated by the Knuth-Bendix completion algorithm.

Groups (group). E_{group} defines group theory by:

$$E_{group} \begin{cases} 1 * x = x \\ I(x) * x = 1 \\ (x * y) * z = x * (y * z) \end{cases}$$

Then the complete system R_{group} is:

$$R_{group} \begin{cases} I(1) = 1 \\ 1 * x = x \\ x * 1 = x \\ I(I(x)) = x \\ I(x) * x = 1 \\ x * I(x) = 1 \\ I(x * y) = I(y) * I(x) \\ (x * y) * z = x * (y * z) \\ I(x) * (x * y) = y \\ x * (I(x) * y) = y \end{cases}$$

Groups'(group'). $E_{group'}$ defines group theory by a right identity and a right inverse:

$$E_{group'} \begin{cases} x * 1 = x \\ x * I(x) = 1 \\ (x * y) * z = x * (y * z) \end{cases}$$

Then the complete system of $E_{group'}$ is R_{group}.

Groups" (group"). $E_{group''}$ defines group theory by Taussky's presentation:

$$E_{group''} \begin{cases} 1 * 1 = 1 \\ x * I(x) = 1 \\ (x * y) * z = x * (y * z) \\ f(1, x, y) = x \\ f(x * y, x, y) = g(x * y, y) \end{cases}$$

Then the complete system $R_{group''}$ is:

$$R_{group''} \begin{cases} I(1) = 1 \\ 1 * x = x \\ x * 1 = x \\ g(1, x) = I(x) \\ f(x, y) = g(x, I(y) * x)I(I(x)) = x \\ I(x) * x = 1 \\ x * I(x) = 1 \\ I(x * y) = I(y) * I(x) \\ (x * y) * z = x * (y * z) \\ I(x) * (x * y) = y \\ x * (I(x) * y) = y \end{cases}$$

(l,r)-Systems (l-r-s). E_{l-r-s} defines (l,r)-system by:

$$E_{l-r-s} \quad \begin{cases} 1 * x = x \\ x * I(x) = 1 \\ (x * y) * z = x * (y * z) \end{cases}$$

Then the complete system R_{l-r-s} is:

$$R_{l-r-s} \quad \begin{cases} I(1) = 1 \\ 1 * x = x \\ I(I(x)) = x * 1 \\ I(x) * 1 = I(x) \\ x * I(x) = 1 \\ I(x * y) = I(y) * I(x) \\ (x * y) * z = x * (y * z) \\ I(x) * (x * y) = y \\ x * (I(x) * y) = y \end{cases}$$

Central Groupoids (c-group). $E_{c-group}$ defines central groupoids theory by:

$$E_{c-group} \quad \begin{cases} (x * y) * (y * z) = y \end{cases}$$

Then the complete system $R_{c-group}$ is:

$$R_{c-group} \quad \begin{cases} (x * y) * (y * z) = y \\ x * ((x * y) * z) = x * y \\ (z * (x * y)) * y) = x * y \end{cases}$$

Reverse (rev). E_{rev} defines the reverse of a list:

$$E_{rev} \quad \begin{cases} append(nil, y) = y \\ append(cons(x, y), z) = cons(x, append(y, z)) \\ reverse(nil) = nil \\ reverse(cons(x, y)) = append(reverse(y), cons(x, nil)) \\ reverse(reverse(x)) = x \end{cases}$$

Then the complete system R_{rev} is:

$$R_{rev} \quad \begin{cases} append(nil, y) = y \\ append(cons(x, y), z) = cons(x, append(y, z)) \\ reverse(nil) = nil \\ reverse(reverse(x)) = x \\ reverse(cons(x, y)) = append(reverse(y), cons(x, nil)) \\ reverse(append(x, cons(y, nil))) = cons(y, reverse(x)) \end{cases}$$

Bibliography

[1] L. Bachmair and N. Dershowitz, Commutation, transformation, and termination, *Lecture Notes in Comput. Sci. Vol. 230* (Springer-Verlag, 1986) 5-20.

[2] H. P. Barendregt, *The lambda calculus, its syntax and semantics* (North-Holland, 1981).

[3] J. A. Bergstra and J. W. Klop, Conditional rewrite rules: Confluence and termination, *J. Comput. and Syst. Sci. 32* (1986) 323-362.

[4] G. Berry and J. J. Lévy, Minimal and optimal computations of recursive programs, *J. ACM 26* (1979) 148-175.

[5] R. V. Book, Confluent and other types of Thue systems, *J. ACM 29* (1982) 171-182.

[6] V. Breazu-Tannen, Combining algebra and higher-order types, *Proc. of the third IEEE Symp. on Logic in Computer Science* (1988) 82-90.

[7] V. Breazu-Tannen and J. Gallier, Polymorphic rewriting conserves algebraic strong normalization and confluence, *Lecture Notes in Comput. Sci. 372* (Springer-Verlag, 1989) 137-150.

[8] B. Buchberger and R. Loos, Algebraic simplification, in: B. Buchberger, G. E. Collins and R. Loos, eds., *Computer Algebra Symbolic and Algebraic Computation* (Springer-Verlag, 1983) 11-43.

[9] R. M. Burstall and J. Darlington, A transformation system for developing recursive programs, *J. ACM 24* (1977) 44-67.

[10] R. M. Burstall, D. B. MacQueen and D. T. Sannella, HOPE: An experimental applicative language, *Proc. of the 1st International Conference on Lisp* (1980) 136-143.

[11] J. Christian, High-performance permutative completion, *MCC Technical Report ACT-AI-303-89* (1989).

[12] A. Church, *The calculi of lambda conversion* (Princeton University Press, 1941).

[13] A. Church and J. B. Rosser, Some properties of conversion, *Trans. Amer. Math. Soc. 39* (1936) 472-482.

[14] H. Comon, Sufficient completeness, term rewriting systems and "anti-unification", *Lecture Notes in Comput. Sci. 230* (Springer-Verlag, 1986) 128-140.

[15] H. B. Curry and R. Feys, *Combinatory logic I* (North-Holland, 1958).

[16] H. B. Curry, J. R. Hindley and J. P. Seldin, *Combinatory logic II* (North-Holland, 1972).

[17] N. Dershowitz and Z. Manna, Proving termination with multiset orderings, *C. ACM 22* (1979) 465-476.

[18] N. Dershowitz, Termination of linear rewriting systems: Preliminary version, *Lecture Notes in Comput. Sci. 115* (Springer-Verlag, 1981) 448-458.

[19] N. Dershowitz. Termination, *Lecture Notes in Comput. Sci. 220* (Springer-Verlag, 1985) 180-224.

[20] N. Dershowitz, Computing with rewriting systems, *Information and Control 65* (1985) 122-157.

[21] N. Dershowitz and D. A. Plaisted, Equational programming, in: J. E. Hayes, D. Michie, and J. Richards, eds., *Machine Intelligence 11* (Oxford 1987) 21-56.

[22] N. Dershowitz and J. P. Jouannaud, Rewrite Systems, *to appear in*: J. V. Leeuwen, ed., *Handbook of Theoretical Computer Science* (North-Holland).

[23] D. J. Dougherty, Adding algebraic rewriting to untyped lambda calculus, *Preliminary Draft*, Wesleyan University (1989).

[24] P. J. Downey and R. Sethi, Correct computation rules for recursive languages, *SIAM J. Comput. 5* (1976) 378-401.

[25] L. Fribourg, SLOG: A logic programming language interpreter based on clausal superposition and rewriting, *Proc. of the 1985 Symp. on Logic Programming* (1985) 172-184.

[26] L. Fribourg, A strong restriction of the inductive completion procedure, *J. Symbolic Computation 8* (1989) 253-276.

[27] K. Futatsugi and Y. Toyama, Term rewriting systems and their applications: A survey, *J. IPS Japan 24 (2)* (1983) 133-146, *in Japanese.*

[28] H. Ganzinger and R. Giegerich, A note on termination in combinations of heterogeneous term rewriting systems, *EATCS Bulletin 31* (1987) 22-28.

[29] A. Geser, H. Hussmann, and A. Mück, A compiler for a class of conditional term rewriting systems, *Lecture Notes in Comput. Sci. 308* (Springer-Verlag, 1988) 84-90.

[30] J. A. Goguen, J. W. Thatcher and E. G. Wagner, An initial algebra approach to the specification, correctness, and implementation of abstract data types,

in: R. Yeh, ed., *Current Trends in Programming Methodology 4*, (Prentice-Hall, 1978) 80-149.

[31] J. A. Goguen, How to prove algebraic inductive hypotheses without induction, with applications to the correctness of data type implementation, *Lecture Notes in Comput. Sci. 87* (Springer-Verlag, 1980) 356-373.

[32] J. A. Goguen and J. Meseguer, EQLOG: Equality, types and generic modules for logic programming, in: D. DeGroot and G. Lindstrom, eds., *Logic Programming: Functions, relations and equations* (Prentice-Hall 1986) 295-363.

[33] J. V. Guttag and J. J. Horning, The algebraic specification of abstract data types, *Acta Informatica 10* (1978) 27-52.

[34] J. R. Hindley, The Church-Rosser property and a result in combinatory logic, Ph.D. Thesis, Univ. of Newcastle-upon-Tyne (1964).

[35] J. Hsiang, Refutational theorem proving using term-rewriting systems, *Artificial Intelligence 25* (1985) 255-300.

[36] J. Hsiang. Private communication (July 28, 1986).

[37] G. Huet, Confluent reductions: Abstract properties and applications to term rewriting systems, *J. ACM 27* (1980) 797-821.

[38] G. Huet and J. M. Hullot, Proofs by induction in equational theories with constructors, *J. Comput. and Syst. Sci. 25* (1982) 239-266.

[39] G. Huet and D. C. Oppen, Equations and rewrite rules: A survey, in: R.V. Book, ed., *Formal languages: perspectives and open problems*, (Academic Press, 1980) 349-405.

[40] T. Ida, A. Aiba and Y. Toyama, T: A simple reduction language based on combinatory term rewriting, in: K. Fuchi and M. Nivat, eds., *Programming of Future Generation Computer I* (North-Holland, 1988) 217-236.

[41] Y. Inagaki and T. Naoi, Term rewriting systems and functional programming, *J. IPS Japan 29 (8)* (1988) 829-835, *in Japanese.*

[42] K. Inoue, H. Seki, K. Taniguchi, and T. Kasami, Compiling and optimizing methods for the functional language ASL/F, *Science of Computer Programming 7-3* (1986) 297-312.

[43] T. Ito, Mathematical theory of programs and its applications (1) $\sim$ (4), *J. IPS Japan 22 (7)* (1981) 681-691, *J. IPS Japan 22 (9)* (1981) 884-895, *J. IPS Japan 22 (11)* (1981) 1069-1079, *J. IPS Japan 23 (2)* (1982) 147-157, *in Japanese.*

[44] J. P. Jouannaud and H. Kirchner, Completion of a set of rules modulo a set of equations, *SIAM J. Comput. 15* (1986) 1155-1194.

[45] J. P. Jouannaud and E. Kounalis, Automatic proofs by induction in theories without constructors, *Information and Computation 82* (1989) 1-33.

[46] J. P. Jouannaud and M. Munoz, Termination of a set of rules modulo a set of equations, *Lecture Notes in Comput. Sci. Vol. 170* (Springer-Verlag, 1984) 175-193.

[47] S. Kaplan, A compiler for conditional term rewriting systems, *Lecture Notes in Comput. Sci. 256* (Springer-Verlag, 1987) 25-41.

[48] S. Kaplan, Conditional rewrite rules, *Theoretical Comput. Sci. 33* (1984) 175-193.

[49] D. Kapur and P. Narendran, An equational approach to theorem proving in first-order predicate calculus, *General Electric Corporate Research Development Report, No.84CRD322* (1985).

[50] D. Kapur, P. Narendran, and H. Zhang, Proof by induction using test sets, *Lecture Notes in Comput. Sci. 230* (Springer-Verlag, 1986) 99-117.

[51] D. Kapur, P. Narendran, and H. Zhang, Complexity of sufficient-completeness, *Lecture Notes in Comput. Sci. 241* (Springer-Verlag, 1986) 426-442.

[52] D. Kapur and H. Zhang, RRL: A rewrite rule laboratory, *Lecture Notes in Comput. Sci. 310* (Springer-Verlag, 1988) 768-769.

[53] H. Kirchner, A general inductive completion algorithm and application to abstract data types, *Lecture Notes in Comput. Sci. 170* (Springer-Verlag, 1985) 282-302.

[54] H. Kirchner, Schematization of infinite sets of rewriting rules: Application to divergence of completion processes, *Lecture Notes in Comput. Sci. 256* (Springer-Verlag, 1987) 180-191.

[55] S. C. Kleene, λ-definability and recursiveness, *Duke Math. J. 2* (1936) 340-353.

[56] J. W. Klop, Combinatory reduction systems, Dissertation, Univ. of Utrecht, 1980.

[57] J. W. Klop and H. P. Barendregt, Private communication (January 19, 1986).

[58] J. W. Klop, Term rewriting systems: A tutorial, *EATCS Bulletin 32* (1987) 143-182.

[59] D. E. Knuth and P. G. Bendix, Simple word problems in universal algebras, in: J. Leech, ed., *Computational problems in abstract algebra* (Pergamon Press, 1970) 263-297.

[60] E. Kounalis, Completeness in data type specifications, *Lecture Notes in Comput. Sci. 204* (Springer-Verlag, 1985) 348-362.

[61] M. Kurihara and I. Kaiji, Modular term rewriting systems: Termination, confluence and strategies, *Preliminary Draft*, Hokkaido University (1988).

[62] P. Lescanne, Computer experiments with the REVE term rewriting system generator, *Proc. 10th ACM Conf. of Principle of Programming Languages* (1983) 99-108.

[63] J. McCarthy, Basis for a mathematical theory of computation, in: P. Braffort and D. Hirschberg, eds., *Computer Programming and Formal Systems* (North-Holland, 1963) 33-70.

[64] A. Middeldorp, A sufficient condition for the termination of the direct sum of term rewriting systems, *Proc. of the 4th IEEE Symp. on Logic in Computer Science* (1989) 394-401.

[65] A. Middeldorp, Modular aspects of properties of term rewriting systems related to normal forms, *Lecture Notes in Comput. Sci. 355* (Springer-Verlag, 1989) 263-277.

[66] A. Middeldorp, Confluence of the disjoint union of conditional term rewriting systems, *Preliminary Draft*, CWI, Amsterdam (1989).

[67] A. Middeldorp, Termination of the disjoint union of conditional term rewriting systems, *Preliminary Draft*, CWI, Amsterdam (1989).

[68] A. Middeldorp, Unique normal forms of the disjoint union of conditional term rewriting systems, *Preliminary Draft*, CWI, Amsterdam (1990).

[69] D. R. Musser, On proving inductive properties of abstract data types, *Proc. 7th ACM Sympo. Principles of Programming Languages* (1980) 154-162.

[70] T. Nipkow and G. Weikum, A decidability result about sufficient-completeness of axiomatically specified abstract data type, *Lecture Notes in Comput. Sci. 145* (Springer-Verlag, 1983) 257-267.

[71] A. Ohsuga and K. Sakai, Metis: A term rewriting system generator, *ICOT Technical Memorandum TM-0226* (1986).

[72] M. J. O'Donnell, Computing in systems described by equations, *Lecture Notes in Comput. Sci. Vol. 58* (Springer-Verlag, 1977).

[73] M. J. O'Donnell, *Equational logic as a programming language* (The MIT Press, 1985).

[74] M. Oyamaguchi, The Church-Rosser property for ground term rewriting systems is decidable, *Theoretical Comput. Sci. 49* (1987) 43-79.

[75] E. Paul, Proof by induction in equational theories with relations between constructors, in: B. Courcelle, ed., *9th Colloquium on trees in algebra and programming* (Cambridge University Press, 1984) 211-225.

[76] E. Paul, On solving the equality problem in theories defined by Horn clauses, *Theoretical Comput. Sci. 44* (1986) 127-153.

[77] D. A. Plaisted, Semantic confluence tests and completion methods, *Information and Control 65* (1985) 182-215.

[78] P. W. Purdom and C. A. Brown, Fast many-to-one matching algorithms, *Lecture Notes in Comput. Sci. 202* (Springer-Verlag, 1985) 407-416.

[79] J. C. Raoult and J. Vuillemin, Operational and semantic equivalence between recursive programs, *J. ACM 27* (1980) 772-796.

[80] J. C. Raoult, On graph rewriting, *Theoretical Comput. Sci. 32* (1984) 1-24.

[81] B. K. Rosen, Tree-manipulating systems and Church-Rosser theorems, *J. ACM 20* (1973) 160-187.

[82] J. B. Rosser, A mathematical logic without variables, *Annals of Math. 36* (1935) 127-150.

[83] M. Rusinowitch, On termination of the direct sum of term rewriting systems, *Inform. Process. Lett. 26* (1987) 65-70.

[84] M. Sakai, T. Sakabe, and Y. Inagaki, Direct implementation system of algebraic specifications of abstract data types, *Computer Software 4-4* (Iwanami, 1987) 16-27, *in Japanese*.

[85] M. Sato and T. Sakurai, QUTE: A functional language based on unification, in: D. DeGroot and G. Lindstrom, eds., *Logic Programming: Functions, relations and equations* (Prentice-Hall 1986) 131-155.

[86] M. Schönfinkel, Über die bausteine der mathematischen logik, *Math. Annalen 92* (1924) 305-316. Translated as: On the building blocks of mathematical logic, in: J. V. Heyenoort, ed., *From Frege to Gödel* (Harvard Univ. Press, 1967) 355-366.

[87] J. Staples, Church-Rosser theorem for replacement systems, *Lecture Notes in Mathematics 450* (Springer-Verlag, 1975) 291-307.

[88] Y. Sugiyama, K. Taniguchi, and T. Kasami, A specification defined as as extension of a base algebra, *Trans. IECE Japan, J64-D, 4* (1981) 324-331, *in Japanese*.

[89] J. J. Thiel, Stop losing sleep over incomplete data type specifications, *Proc. 11th ACM Sympo. Principles of Programming Languages* (1984) 76-82.

[90] A. Togashi and S. Noguchi, Some decision problems and their time complexity for term rewriting systems, *Trans. IECE Japan, J66-D* (1983) 1177-1184, *in Japanese*.

[91] S. Tomura and K. Futatsugi, Transformation system from term rewriting systems into LISP programs, *IEICE (the Institute of Electronics, Information and Communication Engineers) technical report SS86-11* (1986) 15-20, *in Japanese*.

[92] Y. Toyama, On commutativity of term rewriting systems, *Trans. IECE Japan, J66-D* (1983) 1370-1375, *in Japanese*.

[93] Y. Toyama, On equivalence transformations for term rewriting systems, *Lecture Notes in Computer Science 220* (Springer-Verlag, 1985) 44-61.

[94] Y. Toyama, How to prove equivalence of term rewriting systems without induction, *Lecture Notes in Computer Science 230* (Springer-Verlag, 1986) 118-127.

[95] Y. Toyama, On the Church-Rosser property for the direct sum of term rewriting systems, *J. ACM 34* (1987) 128-143.

[96] Y. Toyama, Counterexamples to termination for the direct sum of term rewriting systems, *Inform. Process. Lett. 25* (1987) 141-143.

[97] Y. Toyama, Confluenct term rewriting systems with membership conditions, *Lecture Notes in Computer Science 308* (Springer-Verlag, 1988) 128-141.

[98] Y. Toyama, Commutativity of term rewriting systems, in: K. Fuchi and L. Kott, eds., *Programming of Future Generation Computer II* (North-Holland, 1988) 393-407.

[99] Y. Toyama, J. W. Klop, H. P. Barendregt, Termination for the direct sum of left-linear term rewriting systems: Preliminary draft, *Lecture Notes in Comput. Sci. 355* (Springer-Verlag, 1989) 477-491.

[100] Y. Toyama, J. W. Klop, H. P. Barendregt, Termination for the direct sum of left-linear complete term rewriting systems, *CWI Report CS-R8923* (1989), *submitted to J. ACM.*

[101] Y. Toyama, Membership conditional term rewriting systems, *Trans. IEICE Japan, E72* (1989) 1224-1229.

[102] Y. Toyama, Fast Knuth-Bendix completion with a term rewriting system compiler, *Inform. Process. Lett. 32* (1989) 325-328.

[103] Y. Toyama, How to prove equivalence of term rewriting systems without induction, *to appear in Theoretical Comput. Sci. 78* (1991).

[104] D. A. Turner, The semantic elegance of applicative languages, *Proc. ACM Symp. on Functional Programming Languages and Computer Architecture* (1981) 85-92.

[105] D. A. Turner, Miranda: A non-strict functional language with poloimorphic types, *Lecture Notes in Comput. Sci. 201* (Springer-Verlag, 1985) 1-16.

[106] J. Vuillemin, Correct and optimal implementations of recursion in a simple programming language, *J. Compt. and Syst. Sci. 9* (1974) 332-354.

[107] J. Vuillemin, Syntaxe, sémantique et axiomatique d'un langage de programmation simple, *Th. de Université de Paris VII* (1974)

Paper List

1. Y. Toyama, On the Church-Rosser property for the direct sum of term rewriting systems, *J. ACM 34* (1987) 128-143.

2. Y. Toyama, Counterexamples to termination for the direct sum of term rewriting systems, *Inform. Process. Lett. 25* (1987) 141-143.

3. Y. Toyama, J.W. Klop, H.P. Barendregt, Termination for the direct sum of left-linear complete term rewriting systems, *CWI Report CS-R8923* (1989), *submitted to J. ACM.*

4. Y. Toyama, J.W. Klop, H.P. Barendregt, Termination for the direct sum of left-linear term rewriting systems: Preliminary draft, *Proc. of the 3rd International Conference on Rewriting Techniques and Applications, Lecture Notes in Comput. Sci. 355* (Springer-Verlag, 1989) 477-491.

Chapter 3

5. Y. Toyama, Commutativity of term rewriting systems, in: K. Fuchi and L. Kott, eds., *Programming of Future Generation Computer II* (North-Holland, 1988) 393-407.

6. Y. Toyama, On commutativity of term rewriting systems, *Trans. IECE Japan, J66-D* (1983) 1370-1375, *in Japanese.*

Chapter 4

7. Y. Toyama, How to prove equivalence of term rewriting systems without induction, *to appear in Theoretical Comput. Sci. 78* (1991).

8. Y. Toyama, How to prove equivalence of term rewriting systems without induction, *Proc. of the 8th International Conference on Automated Deduction, Lecture Notes in Computer Science 230* (Springer-Verlag, 1986) 118-127.

9. Y. Toyama, On equivalence transformations for term rewriting systems, *Proc. of RIMS Symposia on Software Science and Engineering II, Lecture Notes in Computer Science 220* (Springer-Verlag, 1985) 44-61.

Chapter 5

10. Y. Toyama, Membership conditional term rewriting systems, *Trans. IEICE Japan, E72* (1989) 1224-1229.

11. Y. Toyama, Confluenct term rewriting systems with membership conditions, *Proc. of the 1st International Workshop on Conditional Term Rewriting Systems, Lecture Notes in Computer Science 308* (Springer-Verlag, 1988) 128-141.

Others

12. Y. Toyama, Fast Knuth-Bendix completion with a term rewriting system compiler, *Inform. Process. Lett. 32* (1989) 325-328.

13. T. Ida, A. Aiba and Y. Toyama, T: A simple reduction language based on combinatory term rewriting, in: K. Fuchi and M. Nivat, eds., *Programming of Future Generation Computer I* (North-Holland, 1988) 217-236.

14. K. Futatsugi and Y. Toyama, Term rewriting systems and their applications: A survey, *J. IPS Japan 24 (2)* (1983) 133-146, *in Japanese*.

15. Y. Toyama, A comment on call by need, *J. IPS Japan 25 (3)* (1984) 249-251, *in Japanese*.

16. K. Fuchi, T. Kurokawa, Y. Toyama, etc., *New Generation Programming* (Kyorichu, 1986), *in Japanese*.

17. Y. Toyama, Greatest common divisor: Euclid's algorithms for universal algebra, polynomial ideal, and automated theorem proving, *to appear in*: T. Ida, ed., *New Programming Paradigm II* (Kyorichu, 1990), *in Japanese*.

18. T. Sugawara and Y. Toyama, Theory of inductive inference: A tutorial, *J. Instrument and Control Engineers, 25* (1986) 781-786, *in Japanese*.

19. Y. Toyama and M. Kimura, On Learning automata in non-stationary random environments, *Trans. IECE Japan, J60-D* (1977) 1085-1092, *in Japanese*.